# Computational Logic

Recall of First-Order Logic

Damiano Zanardini

UPM European Master in Computational Logic (EMCL)
School of Computer Science
Technical University of Madrid
damiano@fi.upm.es

Academic Year 2009/2010

## Syntax of a first-order language

### An *alphabet* $\mathcal{A}$ consists of

- *variable* symbols: $x$, $y$, $z$, $w$, $x'$, ...
- *function* symbols: $f(\_)$, $g(\_,\_)$, ... (*arity* = no. of args: $f/1$, $g/2$)
- *predicate* symbols: $p(\_)$, $q(\_,\_)$, ..., $= /2$
- *connectives*: $\neg$, $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$
- *quantifiers*: $\forall$, $\exists$

---

- *constants* ($a$, $b$, $c$, ...) = 0-arity functions
- *propositions* = 0-arity predicates

### Metalanguage

$F$ and $G$ denote arbitrary formulæ

## Syntax of a first-order language

- variable symbols: $x$, $y$, $z$
- constant (0-ary functions) symbols: $a$, $b$, $c$, $tom$, 0, 1
- function symbols: $f/1$, $g/2$
- predicate symbols: $p/0$, $q/2$, $cat/1$, $+/3$

### Terms

- a variable is a term
- if $t_1, ..., t_n$ are terms and $f$ is an $n$-ary ($n \geq 0$, thus including constants) function symbol, then $f(t_1, ..., t_n)$ is a term

### Examples

$$a \quad f(tom) \quad f(1 \quad a() \quad a(1) \quad f(2) \quad g(g, g(1)) \quad q(0, f(1))$$
$$a + y \quad g(1, f(a)) \quad g(0c) \quad cat(cat(0)) \quad +(a, f(z), c) \quad f(f(f(x)))$$

## Syntax of a first-order language

- variable symbols: $x$, $y$, $z$
- constant (0-ary functions) symbols: $a$, $b$, $c$, $tom$, 0, 1
- function symbols: $f/1$, $g/2$
- predicate symbols: $p/0$, $q/2$, $cat/1$, $+/3$

### Atoms

- if $t_1, ..., t_n$ are terms and $p$ is an $n$-ary ($n \geq 0$) predicate symbol, then $p(t_1, ..., t_n)$ is an atom

### Examples

$$cat(g(x,y)) \quad p \quad q \quad q(\forall x p(x), \forall x p(f(x))) \quad q(p,p) \quad f(q(a,a))$$
$$q(a,f(1)) \quad +(a,f(z),c) \quad q(0,,z) \quad cat(a,1) \quad cat(cat(f(1)))$$

## Syntax of a first-order language

- variable symbols: $x$, $y$, $z$
- constant (0-ary functions) symbols: $a$, $b$, $c$, $tom$, 0, 1
- function symbols: $f/1$, $g/2$
- predicate symbols: $p/0$, $q/2$, $cat/1$, $+/3$

### Formulæ

- an atom is a formula
- if $F$ and $G$ are formulæ, and $x$ is a variable, then $\neg F$, $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$, $(F \leftrightarrow G)$, $\forall x F$ and $\exists x F$ are also formulæ

### Examples

$$(p \rightarrow \neg q(a, f(x))) \qquad p\wedge \qquad \exists z f(1) \qquad \forall a(q(a,0) \leftrightarrow g(0,a))$$
$$\forall p \qquad (\neg cat(a) \wedge (\forall x p \vee \exists y\, cat(y))) \qquad q(a,b) \leftrightarrow tom$$

## Syntax of a first-order language

### Literals

A *literal* is an atom or the negation of an atom

- $p, \;\; \neg p, \;\; q(a, f(1)), \;\; \neg q(a, f(1)), \;\; cat(g(x, y)), \;\; \neg cat(g(x, y))$

### Precedence

- parentheses give an *order* between operators, but can make a formula quite unreadable
- we can use an order of precedence between operators in order to remove some parentheses without introducing *ambiguity*

$$((p \wedge \neg q) \rightarrow (p \vee r)) \quad \rightsquigarrow \quad p \wedge \neg q \rightarrow p \vee r \quad \rightsquigarrow \quad \begin{array}{c} p \wedge (\neg q \rightarrow p) \vee r \\ ((p \wedge \neg q) \rightarrow p) \vee r \\ (p \wedge \neg q) \rightarrow (p \vee r) \\ p \wedge (\neg q \rightarrow p \vee r) \end{array}$$

## Syntax of a first-order language

### Literals

A *literal* is an atom or the negation of an atom

- $p$, $\neg p$, $q(a, f(1))$, $\neg q(a, f(1))$, $cat(g(x, y))$, $\neg cat(g(x, y))$

### Precedence

- parentheses give an *order* between operators, but can make a formula quite unreadable
- we can use an order of precedence between operators in order to remove some parentheses without introducing *ambiguity*

$$((p \land \neg q) \rightarrow (p \lor r)) \quad \rightsquigarrow \quad p \land \neg q \rightarrow p \lor r \quad \rightsquigarrow \quad \begin{array}{c} p \land (\neg q \rightarrow p) \lor r \\ ((p \land \neg q) \rightarrow p) \lor r \\ (p \land \neg q) \rightarrow (p \lor r) \\ p \land (\neg q \rightarrow p \lor r) \end{array}$$

## Syntax of a first-order language

### Literals

A *literal* is an atom or the negation of an atom

- $p, \;\; \neg p, \;\; q(a, f(1)), \;\; \neg q(a, f(1)), \;\; cat(g(x, y)), \;\; \neg cat(g(x, y))$

### Precedence

- parentheses give an *order* between operators, but can make a formula quite unreadable
- we can use an order of precedence between operators in order to remove some parentheses without introducing *ambiguity*
- $\{\neg, \forall, \exists\}$ higher precedence than $\{\wedge, \vee\}$
- $\{\wedge, \vee\}$ higher precedence than $\{\rightarrow, \leftrightarrow\}$

☞ the *scope* of $\exists$ in $\exists y p(y) \wedge q(y)$ is only $p(y)$, not the whole formula

## Syntax of a first-order language

### Free and bounded variable occurrences

- an occurrence of the variable $x$ is *bounded* in $F$ if it is in the scope of a quantifier $\forall x$ or $\exists x$

$$\exists x(p(1, x, y) \wedge q(x)) \wedge q(x)$$

- otherwise, it is said to be *free*

$$\exists x(p(1, x, y) \wedge q(x)) \wedge q(x)$$

- a formula if *closed* if it contains no free occurrences

☞ *occurrences*, not variable symbols, can be free or bounded

## Syntax of a first-order language

### Substitution

- $F(x)$ denotes a formula where $x$ occurs free somewhere
- $F(x/t)$ denotes a formula where every free occurrence of $x$ has been replaced by a term $t$

☞ provided $x$ does not occur free in the scope of any $\forall y$ or $\exists y$ for $y$ occurring in $t$

$F \equiv s(x) \land (\forall y(p(x) \to q(y)))$ $F(x/f(y,y))$ cannot be done

☞ $\forall x p(x)$ is the same as $\forall y p(y)$ (general result: $\forall x F(x) \leftrightarrow \forall y F(x/y)$)

☞ if you are not sure about the name of variables, it is never a mistake to *rename* all the bounded occurrences of a variable

## Syntax of a first-order language

### Alternative notation

| | | | |
|---|---|---|---|
| $\wedge$ | $\&$ | $\rightarrow$ | $\Rightarrow,\ \supset$ |
| $\leftrightarrow$ | $\Leftrightarrow,\ \equiv$ | $\exists xF$ | $\exists x.\ F,\ \exists x \mid F$ |
| $\forall xF$ | $\forall x.\ F,\ \forall x \mid F$ | $p, q, r$ | $P, Q, R$ |

### More than first-order

- *second-order logic*: it allows quantification over functions and predicates (e.g., *mathematical induction*)

$$\forall p(p(0) \wedge \forall k(p(k) \rightarrow p(s(k))) \rightarrow \forall np(n))$$

- *higher-order logic* allows quantification over functions and predicates of any order (as in *functional programming*)

## Semantics of a first-order language

### Interpretations

An *interpretation* $\mathcal{I}$ is a pair $(D, I)$, where $D \neq \emptyset$ is a set (the *domain* of the universe) and $I$ maps symbols to *individuals* or *functions*

- constants: $I(a) = d \in D$
- variables: $I(x) = d \in D$
- functions: $I(f/n) = \mathcal{F} : D^n \mapsto D$
  - $I(f(t_1, ..., t_n)) = \mathcal{F}(I(t_1), ..., I(t_n)) = \mathcal{F}(d_1, ..., d_n) \in D$
- predicates: $I(p/n) = \mathcal{P} : D^n \mapsto \{\mathbf{t}, \mathbf{f}\}$
  - $I(p(t_1, ..., t_n)) = \mathcal{P}(I(t_1), ..., I(t_n)) = \mathcal{P}(d_1, ..., d_n) \in \{\mathbf{t}, \mathbf{f}\}$

☞ an interpretation assigns an element of $D$ to any term, and a *truth value* to any predicate applied to terms

- $\mathcal{P}$ is an $n$-ary *relation* $\mathcal{R}$: $\mathcal{P}(d_1, ..., d_n) = \mathbf{t}$ iff $\langle d_1, ..., d_n \rangle \in \mathcal{R}$

## Semantics of a first-order language

### Evaluation of a formula

Assigning a truth value to a formula, according to:

- the chosen interpretation of constants, functions and predicates
- the rules for evaluation (see also *truth tables*)

| | | |
|---|---|---|
| $I(\neg F) = \mathbf{t}$ | iff | $I(F) = \mathbf{f}$ |
| $I(F \wedge G) = \mathbf{t}$ | iff | $I(F) = I(G) = \mathbf{t}$ |
| $I(F \vee G) = \mathbf{f}$ | iff | $I(F) = I(G) = \mathbf{f}$ |
| $I(F \rightarrow G) = \mathbf{f}$ | iff | $I(F) = \mathbf{t}$ and $I(G) = \mathbf{f}$ |
| $I(F \leftrightarrow G) = \mathbf{t}$ | iff | $I(F) = I(G)$ |
| $I(\forall x F(x)) = \mathbf{t}$ | iff | $I(F(x/c)) = \mathbf{t}$ * for every constant $c$ |
| $I(\exists x F(x)) = \mathbf{t}$ | iff | $I(F(x/c)) = \mathbf{t}$ * for at least one constant $c$ |

* it is required that every element $d$ of $D$ is denoted by at least one constant

- $I$ (instead of $\mathcal{I}$) will often denote an interpretation when $D$ is clear

## Semantics of a first-order language

### Example (propositional): $(p \rightarrow q) \wedge (q \rightarrow r) \rightarrow r$

- first interpretation: $I'(p) = \mathbf{f}$, $I'(q) = \mathbf{f}$, $I'(r) = \mathbf{f}$

$$\begin{array}{ccccccccc}
(\mathbf{f} & \rightarrow & \mathbf{f}) & \wedge & (\mathbf{f} & \rightarrow & \mathbf{f}) & \rightarrow & \mathbf{f} \\
 & \mathbf{t} & & \wedge & & \mathbf{t} & & \rightarrow & \mathbf{f} \\
 & & & \mathbf{t} & & & & \rightarrow & \mathbf{f} \\
 & & & & \mathbf{f} & & & &
\end{array}$$

## Semantics of a first-order language

**Example (propositional): $(p \rightarrow q) \wedge (q \rightarrow r) \rightarrow r$**

- second interpretation: $I''(p) = \mathbf{f}$, $I''(q) = \mathbf{f}$, $I''(r) = \mathbf{t}$

$$\begin{array}{ccccccccccc}
(\mathbf{f} & \rightarrow & \mathbf{f}) & \wedge & (\mathbf{f} & \rightarrow & \mathbf{t}) & \rightarrow & \mathbf{t} \\
 & \mathbf{t} & & \wedge & & \mathbf{t} & & \rightarrow & \mathbf{t} \\
 & & & \mathbf{t} & & & & \rightarrow & \mathbf{t} \\
 & & & & \mathbf{t} & & & &
\end{array}$$

## Semantics of a first-order language

### Example (propositional): $(p \rightarrow q) \wedge (q \rightarrow r) \rightarrow r$

- this example only needs truth tables, for all possible interpretations

| $p$ | $q$ | $r$ | $p \rightarrow q$ | $q \rightarrow r$ | $(p \rightarrow q) \wedge (q \rightarrow r)$ | $(p \rightarrow q) \wedge (q \rightarrow r) \rightarrow r$ |
|---|---|---|---|---|---|---|
| **t** | **t** | **t** | **t** | **t** | **t** | **t** |
| **t** | **t** | **f** | **t** | **f** | **f** | **t** |
| **t** | **f** | **t** | **f** | **t** | **f** | **t** |
| **t** | **f** | **f** | **f** | **t** | **f** | **t** |
| **f** | **t** | **t** | **t** | **t** | **t** | **t** |
| **f** | **t** | **f** | **t** | **f** | **f** | **t** |
| **f** | **f** | **t** | **t** | **t** | **t** | **t** |
| **f** | **f** | **f** | **t** | **t** | **t** | **f** |

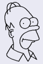
## Semantics of a first-order language

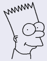
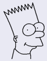
**Example (first-order): $\forall x(m(a,x) \wedge p(x)) \rightarrow \forall y q(s(y))$**

- first interpretation: $D = \{0, 1, 2, 3, ..\}$
  - $I(a) = 0$
  - $I(s(x)) = \mathcal{S}(I(x)) =$ the successor of $I(x)$
  - $p(x)$ means that $x$ is even
  - $q(x)$ means that $x$ is odd
  - $m(x,y)$ means that $x < y$
- $\mathcal{I}$ evaluates the formula to **t** (try it!)

## Semantics of a first-order language

### Example (first-order): $\forall x(m(a,x) \land p(x)) \rightarrow \forall y q(s(y))$

- second interpretation: $D = \{$ , , $\}$, $I(a) =$ 

| $x$ | $s(x)$ |
|---|---|
| | |
| |  |
| |  |

| $x$ | $p(x)$ |
|---|---|
| | **t** |
| | **t** |
| | **t** |

| $x$ | $q(x)$ |
|---|---|
| | **t** |
| | **f** |
| | **t** |

| $m$ $y$ / $x$ | | | |
|---|---|---|---|
| | **t** | **t** | **t** |
| | **f** | **f** | **t** |
| | **t** | **f** | **f** |

- and this evaluates to **f**

## Semantics of a first-order language

### Satisfiable formulæ

An interpretation $\mathcal{I} = (D, I)$ *satisfies* a formula $F$ on $D$ iff $I(F) = \mathbf{t}$ (also written $\mathcal{I}(F) = \mathbf{t}$). In this case, $\mathcal{I}$ is a *model* of $F$

- $F$ is *satisfiable* (written $SAT(F)$) iff it has at least one model
- $F$ is *unsatisfiable* (written $UNSAT(F)$) iff it has no models
  - that is, all interpretations are *countermodels*
- $F$ is *valid* (written $VAL(F)$) iff every interpretation is a model
  - this is denoted by $\models F$, and amounts to say $UNSAT(\neg F)$

With a set of formulæ $\{F_1, .., F_n\}$:

- $(D, I)$ satisfies $\{F_1, .., F_n\}$ iff $I(F_i) = \mathbf{t}$ on $D$ for every $i$
- $\{F_1, .., F_n\}$ is satisfiable iff there is such an interpretation

### Example: $\forall x(m(a, x) \wedge p(x)) \rightarrow \forall y q(s(y))$

- this formula is satisfiable but not valid

## Logical consequence

### Logical consequence

Given a set of formulæ $\Gamma = \{F_1, .., F_n\}$ and a formula $G$ over the same language, $G$ is a *logical consequence* of $\Gamma$ (written $\Gamma \models G$) iff every interpretation satisfying $\Gamma$ also satisfies $G$, or, equivalently, there is no interpretation which satisfies $\Gamma$ but not $G$

### Important (in some sense, it is a matter of convenience)

$$\{F_1, .., F_n\} \models G \qquad \text{iff} \qquad \models (F_1 \wedge .. \wedge F_n) \rightarrow G$$

### To decide $\Gamma \models G$ can be very hard

We have to take all models of $\Gamma$ and verify that they all satisfy $G$, or find a counterexample

## Logical consequence

**Example:** $\{p \rightarrow (q \rightarrow r),\ p \wedge q\} \models r$

☞ equivalent to $\models ((p \rightarrow (q \rightarrow r)) \wedge (p \wedge q)) \rightarrow r$

| $p$ | $q$ | $r$ | $p \rightarrow (q \rightarrow r)$ | $p \wedge q$ | $(p \rightarrow (q \rightarrow r)) \wedge (p \wedge q)$ | $G$ |
|---|---|---|---|---|---|---|
| **t** | **t** | **t** | **t** | **t** | **t** | **t** |
| **t** | **t** | **f** | **f** | **t** | **f** | **t** |
| **t** | **f** | **t** | **t** | **f** | **f** | **t** |
| **t** | **f** | **f** | **t** | **f** | **f** | **t** |
| **f** | **t** | **t** | **t** | **f** | **f** | **t** |
| **f** | **t** | **f** | **t** | **f** | **f** | **t** |
| **f** | **f** | **t** | **t** | **f** | **f** | **t** |
| **f** | **f** | **f** | **t** | **f** | **f** | **t** |

## Logical consequence

**Example: $\{p \rightarrow (q \rightarrow r),\ p \wedge q\} \models \neg r$**

☞ equivalent to $\models ((p \rightarrow (q \rightarrow r)) \wedge (p \wedge q)) \rightarrow \neg r$

| $p$ | $q$ | $r$ | $p \rightarrow (q \rightarrow r)$ | $p \wedge q$ | $(p \rightarrow (q \rightarrow r)) \wedge (p \wedge q)$ | $G$ |
|---|---|---|---|---|---|---|
| **t** | **t** | **t** | **t** | **t** | **t** | **f** |
| **t** | **t** | **f** | **f** | **t** | **f** | **t** |
| **t** | **f** | **t** | **t** | **f** | **f** | **t** |
| **t** | **f** | **f** | **t** | **f** | **f** | **t** |
| **f** | **t** | **t** | **t** | **f** | **f** | **t** |
| **f** | **t** | **f** | **t** | **f** | **f** | **t** |
| **f** | **f** | **t** | **t** | **f** | **f** | **t** |
| **f** | **f** | **f** | **t** | **f** | **f** | **t** |

## Logical consequence

### Example: $\{\exists x p(x),\ \exists x q(x)\} \models \exists x (p(x) \wedge q(x))$

- $D = \{1, 2, 3, 4, ..\}$
- $p(x)$: $x$ is even
- $q(x)$: $x$ is odd

It is easy to see that this interpretation makes both premises true (indeed, there exist even numbers and there exist odd numbers), but does not satisfy the conclusion (no numbers are both even and odd)

$$I(\exists x p(x)) = \mathbf{t} \quad I(\exists x q(x)) = \mathbf{t} \quad I(\exists x (p(x) \wedge q(x))) = \mathbf{f}$$

Therefore, this deduction is incorrect

### Example: $\{\exists x p(x),\ \exists x q(x)\} \models \exists x (p(x) \vee q(x))$

- this is, of course, a correct deduction (or *valid*, see a few slides above)

## Syntax vs. semantics

### Formal systems

A *proof formal system* consists of:

- a formal language (alphabet and rules for building formulæ)
- a set of *logical axioms* (i.e., valid formulæ, which do not require proof)
- a set of *inference rules* for proving new formulæ
- a definition of proof

### Theories

A *theory* $T$ is a formal system extended with a set $\Gamma$ of *non-logical axioms* (i.e., formulæ taken for granted)

$$T[\Gamma]$$

- if $\Gamma = \emptyset$, then $T$ is the *basic theory* of the formal system

## Syntax vs. semantics

### Proofs

A *proof* of a formula $G$ in a theory $T[\Gamma]$ (written $T[\Gamma] \vdash G$) is a *finite* sequence of formulæ such that

- each formula of the sequence is either
  - a logical or non-logical axiom of the theory; or
  - the result of applying an inference rule to previous formulæ in the sequence
- $G$ is the last formula in the sequence

### Theorems (of a theory)

- a *theorem* of the theory $T[\Gamma]$ is a formula for which there is at least one proof in $T[\Gamma]$

## Syntax vs. semantics

### Proof example: $T[p \rightarrow (q \rightarrow r),\ p \wedge q] \vdash r$

| | | |
|---|---|---|
| ❶ | $p \rightarrow (q \rightarrow r)$ | first premise |
| ❷ | $\neg p \vee (\neg q \vee r)$ | interdefinition of $\rightarrow$, $\neg$ and $\vee$ on ❶ |
| ❸ | $(\neg p \vee \neg q) \vee r$ | associativity on ❷ |
| ❹ | $\neg(p \wedge q) \vee r$ | De Morgan on ❸ |
| ❺ | $p \wedge q \rightarrow r$ | interdefinition of $\rightarrow$, $\neg$ and $\vee$ on ❹ |
| ❻ | $p \wedge q$ | second premise |
| ❼ | $r$ | modus ponens on ❺, ❻ |

### Another approach to prove validity

Instead of looking at all the possible models of a formula, we exploited our knowledge of logical rules

- we also say that $((p \rightarrow (q \rightarrow r)) \wedge (p \wedge q)) \rightarrow r$ is a *tautology*

## Syntax vs. semantics

**Theorem (Validity)**

*Every theorem of $T$ is logically valid: if $T \vdash G$ then $\models G$*

**Theorem (Completeness)**

*In a first-order theory $T$, all valid formulæ are theorems of $T$: if $\models G$ then $T \vdash G$*

- *the rest of the course will be basically about finding such formulæ*

**Theorem (Deduction)**

- $T[F_1, .., F_n] \vdash G \quad \textit{iff} \quad T \vdash (F_1 \wedge .. \wedge F_n) \rightarrow G$

## Syntax vs. semantics

**Theorem (Validity)**

*Every theorem of $T$ is logically valid: if $T \vdash G$ then $\models G$*

**Theorem (Completeness)**

*In a first-order theory $T$, all valid formulæ are theorems of $T$: if $\models G$ then $T \vdash G$*

- *the rest of the course will be basically about finding such formulæ*

**Theorem (Deduction)**

- $T[F_1, .., F_n] \vdash G$ *iff* $T \vdash (F_1 \wedge .. \wedge F_n) \rightarrow G$
- $T[F_1, .., F_n] \vdash G$ *iff* $VAL((F_1 \wedge .. \wedge F_n) \rightarrow G)$

## Syntax vs. semantics

**Theorem (Validity)**

*Every theorem of $T$ is logically valid: if $T \vdash G$ then $\models G$*

**Theorem (Completeness)**

*In a first-order theory $T$, all valid formulæ are theorems of $T$: if $\models G$ then $T \vdash G$*

- *the rest of the course will be basically about finding such formulæ*

**Theorem (Deduction)**

- $T[F_1, .., F_n] \vdash G \quad iff \quad T \vdash (F_1 \wedge .. \wedge F_n) \rightarrow G$
- $T[F_1, .., F_n] \vdash G \quad iff \quad VAL((F_1 \wedge .. \wedge F_n) \rightarrow G)$
- $T[F_1, .., F_n] \vdash G \quad iff \quad UNSAT(F_1 \wedge .. \wedge F_n \wedge \neg G)$

## Syntax vs. semantics

### Completeness vs. Incompleteness

- in theories where Gödel's first incompleteness theorem holds, the completeness theorem (also Gödel's) holds as well
- incompleteness: an *effectively generated* (whose set of axioms is a recursively enumerable set) theory which is powerful enough to express *elementary arithmetic* cannot be both *consistent* (there is no statement so that both the statement and its negation are provable from the axioms) and *complete* (for any statement in the axioms' language, either that statement or its negation is provable from the axioms)
- there is a statement which is true in the theory, but cannot be proven nor disproven (the *Gödel sentence*)
- but such statement is *not* a logical consequence of the theory (i.e., there are *non-standard* models, not isomorphic to the standard one, of the theory where it does not hold)

## What you ABSOLUTELY have to get

### Pay attention to these sentences (do they make sense?)

- the interpretation $\mathcal{I}$ is satisfiable
- there exists a valid model for $F$
- a formula $F$ makes an interpretation $I$ false
- in order to prove that a formula is satisfiable, we need a true interpretation
- a literal is a negated or non-negated term

☞ we are not asking if these sentences are true or false, only if they make sense

## What you ABSOLUTELY have to get

### Names and individuals

Names are only there to denote things

- there is no problem in taking

  $$\mathit{female}\left(\quad\right) \qquad \text{or even} \qquad \mathit{odd}(4)$$

  to be true if we are always consistent about it
- never forget that you are dealing with symbols!

## What you ABSOLUTELY have to get

### Types in first-order logic!

- a function of arity $n$ will *never* have arity $m \neq n$ in the same formal system
- a predicate of arity $n$ will *never* have arity $m \neq n$ in the same formal system
- a function is not a predicate
- a predicate is not a function

### Equality

- the predicate $=/2$ is special in the sense that its meaning is often given for granted in a formal system (i.e., being equal means being the same element of the domain)
- anyway, we could choose to redefine it!

## What you ABSOLUTELY have to get

### Interpretations and imagination

When we want to prove that a formula is not valid, we need *one* interpretation which makes it false

- you can choose *anything* you want as $D$ and $I$
- it's ok if we take *that* function $s/1$ to map 45 to !

## What you ABSOLUTELY have to get

### Implication

- that the left-hand side of an implication is false is enough to say that the implication is true
- that the left-hand side of an implication is false is enough to say that the implication is true
- that the left-hand side of an implication is false is enough to say that the implication is true
- that the left-hand side of an implication is false is enough to say that the implication is true
- that the left-hand side of an implication is false is enough to say that the implication is true
- that the left-hand side of an implication is false is enough to say that the implication is true
- that the left-hand side of an implication is false is enough to say that the implication is true

# What you ABSOLUTELY have to get

## Implication, cont.

- *all red apples are good*

$$\forall x((apple(x) \wedge red(x)) \rightarrow good(x))$$

  - there are apples which are red and good
  - all apples are red and good
  - there are no apples which are red but not good
  - the set of good apples is a superset of the set of red apples
  - whenever an apple is not good, it cannot be red
  - whenever an apple is not good, it must be red

- *there exists a yellow apple which is bad*

$$\exists x(apple(x) \wedge yellow(x) \wedge bad(x))$$

  - whenever an apple is yellow, it is bad
  - there is at least an object which is bad and yellow, and is an apple
  - every time an apple is good, it is not yellow

## What you ABSOLUTELY have to get

### Implication, cont.

- *all red apples are good*

$$\forall x((apple(x) \wedge red(x)) \rightarrow good(x))$$

  - there are apples which are red and good $\rightsquigarrow$ NO
  - all apples are red and good $\rightsquigarrow$ NO
  - there are no apples which are red but not good $\rightsquigarrow$ YES
  - the set of good apples is a superset of the set of red apples $\rightsquigarrow$ YES
  - whenever an apple is not good, it cannot be red $\rightsquigarrow$ YES
  - whenever an apple is not good, it must be red $\rightsquigarrow$ NO

- *there exists a yellow apple which is bad*

$$\exists x(apple(x) \wedge yellow(x) \wedge bad(x))$$

  - whenever an apple is yellow, it is bad $\rightsquigarrow$ NO
  - there is at least an object which is bad and yellow, and is an apple $\rightsquigarrow$ YES
  - every time an apple is good, it is not yellow $\rightsquigarrow$ NO