Logik für Informatiker
Logic for computer scientists

Proof rules for quantifiers

Till Mossakowski, Lutz Schröder

WiSe 2011/12

Universal Elimination ($\forall$ Elim)

$$
\begin{array}{r|l}
 & \forall x\, S(x) \\
 & \vdots \\
\triangleright & S(c)
\end{array}
$$

Existential Introduction (∃ Intro)

$$\begin{array}{l|l} & S(c) \\ & \vdots \\ \triangleright & \exists x\, S(x) \end{array}$$

Example: $\forall$-**Elim** and $\exists$-**Intro**

$\forall x[\mathrm{Cube}(x) \rightarrow \mathrm{Large}(x)]$
$\forall x[\mathrm{Large}(x) \rightarrow \mathrm{LeftOf}(x, b)]$
$\mathrm{Cube}(d)$

$\exists x[\mathrm{Large}(x) \wedge \mathrm{LeftOf}(x, b)]$

Existential Elimination ($\exists$ Elim):

$$
\begin{array}{l|l}
 & \exists x\, S(x) \\
 & \vdots \\
 & \quad \boxed{c}\; S(c) \\
 & \quad \vdots \\
 & \quad Q \\
\triangleright & Q
\end{array}
$$

Where c does not occur outside the subproof where it is introduced.

Example: ∃-Elim

$$\forall x[\text{Cube}(x) \rightarrow \text{Large}(x)]$$
$$\forall x[\text{Large}(x) \rightarrow \text{LeftOf}(x, b)]$$
$$\exists x \ \text{Cube}(x)$$

$$\exists x[\text{Large}(x) \land \text{LeftOf}(x, b)]$$

General Conditional Proof (∀ Intro):

$$
\begin{array}{l|l}
 & \quad \begin{array}{|l} \boxed{c}\ P(c) \\ \hline \quad \vdots \\ Q(c) \end{array} \\
\triangleright & \forall x\, (P(x) \to Q(x))
\end{array}
$$

Where c does not occur outside the subproof where it is introduced.

Example: General Conditional Proof

$\forall x[\mathsf{Cube}(x) \rightarrow \mathsf{Large}(x)]$
$\forall x[\mathsf{Large}(x) \rightarrow \mathsf{LeftOf}(x, b)]$

$\forall x[\mathsf{Cube}(x) \rightarrow \mathsf{LeftOf}(x, b)]$

Universal Introduction ($\forall$ Intro):

$\boxed{c}$

$\vdots$

$P(c)$

$\triangleright \; \forall x\, P(x)$

Where c does not occur outside the subproof where it is introduced.

Prenex normal form (reminder)

$$\exists x \mathrm{Cube}(x) \to \forall y \mathrm{Small}(y)$$

$$\forall x \forall y (\mathrm{Cube}(x) \to \mathrm{Small}(y))$$

Example with multiple quantifiers

$$\exists y[\mathrm{Girl}(y) \land \forall x(\mathrm{Boy}(x) \rightarrow \mathrm{Likes}(x, y))]$$

$$\forall x[\mathrm{Boy}(x) \rightarrow \exists y(\mathrm{Girl}(y) \land \mathrm{Likes}(x, y))]$$

Example: de Morgan's Law

$$\begin{array}{|l} \neg\forall x\, P(x) \\ \hline \exists x\, \neg P(x) \end{array}$$

(is not valid in intuitionistic logic, only in classical logic)

Example: The Barber Paradox

$$\exists z\ \exists x\ [ManOf(x,z) \wedge \forall y\ (ManOf(y,z) \rightarrow (Shave(x,y) \leftrightarrow \neg Shave(y,y)))]$$

$$\bot$$

Induction

Induction is like a chain of dominoes. You need

- the dominoes must be close enough together ⇒ one falling dominoe knocks down the next (*inductive step*)
- you need to knock down the first dominoe (*inductive basis*)

Note: in the inductive step, branching is possible.

Induction

Induction is like a chain of dominoes. You need

- the dominoes must be close enough together $\Rightarrow$ one falling dominoe knocks down the next (*inductive step*)
- you need to knock down the first dominoe (*inductive basis*)

Note: in the inductive step, branching is possible.

Inductive definition: Natural numbers

1. 0 is a natural number.
2. If n is natural number, then $suc(n)$ is a natural number.
3. There is no natural number whose successor is 0.
4. Two different natural numbers have different successors.
5. Nothing is a natural number unless generated by repeated applications of (1) and (2).

Recursive definition of functions

$$\forall y(0 + y = y)$$
$$\forall x \forall y(suc(x) + y = suc(x + y))$$

$$\forall y(0 * y = 0)$$
$$\forall x \forall y(suc(x) * y = (x * y) + y)$$

Formalization of Peano's axioms

1. a constant 0
2. a unary function symbol *suc*
3. $\forall n\ \neg suc(n) = 0$
4. $\forall m \forall n\ suc(m) = suc(n) \rightarrow m = n$
5. $(\Phi(x/0) \wedge \forall n(\Phi(x/n) \rightarrow \Phi(x/suc(n)))) \rightarrow \forall n\ \Phi(x/n)$
   if Φ is a formula with a free variable x, and
   $\Phi(x/t)$ denotes the replacement of x with t within Φ

Inductive proofs

Take $\Phi(x) := \forall y \forall z(x + (y + z) = (x + y) + z)$. Then

$$(\Phi(x/0) \wedge \forall n(\Phi(x/n) \rightarrow \Phi(x/suc(n)))) \rightarrow \forall n\ \Phi(x/n)$$

is just

$$\begin{array}{l}(\forall y \forall z(0 + (y + z) = (0 + y) + z) \\ \quad \wedge \forall n \forall y \forall z\ (n + (y + z) = (n + y) + z \\ \qquad \rightarrow suc(n) + (y + z) = (suc(n) + y) + z)) \\ \quad \rightarrow \forall n \forall y \forall z\ (n + (y + z) = (n + y) + z)\end{array}$$

With this, we can prove $\forall n \forall y \forall z\ (n + (y + z) = (n + y) + z)$

1. The empty list [ ] is a list.
2. If l is a list and n is natural number, then $cons(n, l)$ is a list.
3. Nothing is a list unless generated by repeated applications of (1) and (2).

Note: This needs *many-sorted* first-order logic.
We have two sorts of objects: natural numbers and lists.

Recursive definition of functions over lists

$$
\begin{aligned}
&length([\,]) \;=\; 0\\
&\forall n : Nat\ \forall l : List\ (length(cons(n, l)) \;=\; suc(length(l)))
\end{aligned}
$$

$$
\begin{aligned}
&\forall l : List\ ([\,] \ {+}{+}\ l \;=\; l)\\
&\forall n : Nat\ \forall l_1 : List\ \forall l_2 : List\\
&\qquad (cons(n, l_1) \ {+}{+}\ l_2 \;=\; cons(n, l_1 \ {+}{+}\ l_2))
\end{aligned}
$$

Inductive proofs over lists

$$\forall l_1 : List\ \forall l_2 : List\ \forall l_3 : List$$
$$(\ l_1 \ ++ \ (l_2 \ ++ \ l_3) \ = \ (l_1 \ ++ \ l_2) \ ++ \ l_3\ )$$

$$\forall l_1 : List\ \forall l_2 : List$$
$$(\ length(l_1 \ ++ \ l_2) \ = \ length(l_1) \ + \ length(l_2)\ )$$