First Order Predicate Logic
Formal Reasoning in Special Domains

Wolfgang Schreiner and Wolfgang Windsteiger
Wolfgang.(Schreiner|Windsteiger)@risc.jku.at

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University (JKU), Linz, Austria
http://www.risc.jku.at
Formal Reasoning in Special Domains

We will consider methods for

- reasoning about natural numbers,
- reasoning about program loops,

both of which are based on the principle of induction.
Mathematical Induction

A method to prove statements over the natural numbers.

- **Goal:** prove

\[ \forall x \in \mathbb{N} : F \]

i.e., formula \( F \) holds for all natural numbers.

- **Rule:**

\[
\begin{align*}
K \ldots \vdash F[0/x] & \quad K \ldots \vdash (\forall y \in \mathbb{N} : F[y/x] \to F[y + 1/x]) \\
\hline
K \ldots \vdash \forall x \in \mathbb{N} : F 
\end{align*}
\]

\( F[t/x] \): \( F \) where every free occurrence of \( x \) is replaced by \( t \).

- **Proof Steps:**
  - **Induction base:** prove that \( F \) holds for 0.
  - **Induction hypothesis:** assume that \( F \) holds for new constant \( \overline{x} \).
  - **Induction step:** prove that then \( F \) also holds for \( \overline{x} + 1 \).

  *Often the constant symbol \( x \) itself is chosen rather than \( \overline{x} \).*

Works because every natural number is reachable by a finite number of increments starting from 0.
Example

We prove Gauss’s sum formula

\[ \forall n \in \mathbb{N} : \sum_{i=1}^{n} i = \frac{n \cdot (n+1)}{2} \]

by induction on \( n \):

- **Induction Base:**
  \[ \sum_{i=1}^{0} i = 0 = \frac{0 \cdot (0+1)}{2} \]

- **Induction Hypothesis:**
  \[ \sum_{i=1}^{\bar{n}} i = \frac{\bar{n} \cdot (\bar{n}+1)}{2} \quad (\star) \]

- **Induction Step:**
  \[ \sum_{i=1}^{\bar{n}+1} i = (\bar{n}+1) + \sum_{i=1}^{\bar{n}} i \overset{\star}{=} (\bar{n}+1) + \frac{\bar{n} \cdot (\bar{n}+1)}{2} \]
  \[ = \frac{2 \cdot (\bar{n}+1) + \bar{n} \cdot (\bar{n}+1)}{2} = \frac{(\bar{n}+2) \cdot (\bar{n}+1)}{2} \]
Choice of Induction Variable

We define addition on $\mathbb{N}$ by primitive recursion:

\[
\begin{align*}
x + 0 & := x \\
x + (y + 1) & := (x + y) + 1
\end{align*}
\]

Our goal is to prove the associativity law

\[
\forall x \in \mathbb{N}, y \in \mathbb{N}, z \in \mathbb{N} : x + (y + z) = (x + y) + z
\]

For this purpose, we prove

\[
\forall z \in \mathbb{N} : \forall x \in \mathbb{N}, y \in \mathbb{N} : x + (y + z) = (x + y) + z
\]

by induction on $z$.

Sometimes the appropriate choice of the induction variable is critical.
Choice of Induction Variable

We prove by induction on $z$

\[ \forall z \in \mathbb{N}: \forall x \in \mathbb{N}, y \in \mathbb{N}: x + (y + z) = (x + y) + z \]

- **Induction base**: we prove

  \[ \forall x \in \mathbb{N}, y \in \mathbb{N}: x + (y + 0) = (x + y) + 0 \]

  We prove for arbitrary $x_0, y_0 \in \mathbb{N}$

  \[ x_0 + (y_0 + 0) \overset{(1)}{=} x_0 + y_0 \overset{(1)}{=} (x_0 + y_0) + 0 \]

- **Induction hypothesis (\(*\)**: we assume

  \[ \forall x \in \mathbb{N}, y \in \mathbb{N}: x + (y + z) = (x + y) + z \]

- **Induction step**: we prove

  \[ \forall x \in \mathbb{N}, y \in \mathbb{N}: x + (y + (z + 1)) = (x + y) + (z + 1) \]

  We prove for arbitrary $x_0, y_0 \in \mathbb{N}$

  \[ x_0 + (y_0 + (z + 1)) \overset{(2)}{=} x_0 + ((y_0 + z) + 1) \overset{(2)}{=} (x_0 + (y_0 + z)) + 1 \]

  \[ \overset{(*)}{=} \left( (x_0 + y_0) + z \right) + 1 \overset{(2)}{=} (x_0 + y_0) + (z + 1) \]
Induction with a Different Starting Value

▶ **Goal:** prove

\[ \forall x \in \mathbb{N} : x \geq b \rightarrow F \]

i.e., formula \( F \) holds for all natural numbers greater than or equal to some natural number \( b \).

▶ **Rule:**

\[
K \ldots \vdash F[b/x] \quad K \ldots \vdash (\forall y \in \mathbb{N} : y \geq b \land F[y/x] \rightarrow F[y+1/x])
\]

\[
K \ldots \vdash (\forall x \in \mathbb{N} : x \geq b \rightarrow F)
\]

▶ **Proof Steps:**

▶ *Induction base:* prove that \( F \) holds for \( b \).

▶ *Induction hypothesis:* assume that \( F \) holds for \( x \geq b \).

▶ *Induction step:* prove that then \( F \) also holds for \( x + 1 \).

Induction works with arbitrary starting values.
Example

We prove

\[ \forall n \in \mathbb{N} : n \geq 4 \rightarrow n^2 \leq 2^n \]

- **Induction base:** we show

\[ 4^2 = 16 = 2^4 \]

- **Induction hypothesis:** we assume for \( n \geq 4 \)

\[ n^2 \leq 2^n \quad \text{(\( \star \))} \]

- **Induction step:** we show

\[
(n+1)^2 = n^2 + 2n + 1 \quad \text{\( \leq \)} \quad n^2 + 2n + n = n^2 + 3n \quad \text{\( \leq \)} \quad n^2 + 4n
\]

\[
\leq n^2 + n \cdot n = n^2 + n^2 = 2n^2 \quad \text{\( \leq \)} \quad 2 \cdot 2^n = 2^{n+1} \quad \square
\]
Complete Induction

A generalized form of the induction method.

- Rule:

$$K \ldots \vdash (\forall x \in \mathbb{N} : (\forall y \in \mathbb{N} : y < x \rightarrow F[y/x]) \rightarrow F)$$

$$K \ldots \vdash \forall x \in \mathbb{N} : F$$

- Proof steps:
  - Induction hypothesis: assume that $F$ holds for all $y$ less than $x$.
  - Induction step: prove that $F$ then also holds for $x$.

The induction assumption is applied not only to the direct predecessor.
Example

We take function \( T : \mathbb{N} \to \mathbb{N} \) where

\[
T(n) = \begin{cases} 
0 & \text{if } n = 0 \\
2 \cdot T(n/2) & \text{if } n > 0 \land 2|n \\
1 + 2 \cdot T((n-1)/2) & \text{else}
\end{cases}
\]

and prove by complete induction on \( n \)

\[
\forall n \in \mathbb{N} : T(n) = n
\]

- **Induction hypothesis:**
  \[
  \forall m \in \mathbb{N} : m < n \implies T(m) = m 
  \]  
  \( (*) \)

- **Induction step:**
  - Case \( n = 0 \): we know \( T(n) = T(0) = 0 = n \)
  - Case \( n > 0 \land 2|n \): we know
    \[
    T(n) = 2 \cdot T(n/2) \overset{(*)}{=} 2 \cdot (n/2) = n
    \]
  - Case \( n > 0 \land \neg (2|n) \): we know
    \[
    T(n) = 1 + 2 \cdot T((n-1)/2) \overset{(*)}{=} 1 + 2 \cdot ((n-1)/2) = 1 + (n-1) = n
    \]
Computer Programs

Also the correctness of loop-based programs can be proved by induction.

- We consider loops of form
  \[ \text{for} (i=0; i<n; i++) x = t(x,i); \]

- We want to prove that
  - if a precondition \( P(x) \) holds before the execution of the loop,
  - then a postcondition \( Q(x) \) holds afterwards.

- First we prove by induction that, for all \( i \leq n \), some suitable loop invariant \( I(x, i) \) holds after \( i \) iterations of the loop:
  - \( I \) holds initially, i.e., after 0 iterations:
    \[ P(x) \rightarrow I(x,0) \]
  - If \( I \) holds after \( i < n \) iterations, then it also holds after \( i + 1 \) iterations:
    \[ I(x, i) \land i < n \rightarrow I(t(x, i), i + 1) \]

- It then suffices to prove that at the termination of the loop \( (i = n) \) the invariant implies the postcondition:
  \[ I(x, n) \rightarrow Q(x) \]
Example

- **Program**
  
  $$\textbf{for}(i=0; \ i<n; \ i++) \ x = x+2\cdot i+1;$$

- **Precondition** $P(x) : \iff x = 0$

<table>
<thead>
<tr>
<th>$x$</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>9</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i$</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4=n</td>
</tr>
</tbody>
</table>

- **Postcondition** $Q(x) : \iff x = n^2$

- **Loop invariant** $I(x, i) : \iff x = i^2$
  
  - $P(x) \rightarrow I(x, 0)$
    
    $$x = 0 \rightarrow x = 0^2$$

  - $I(x, i) \land i < n \rightarrow I(x+2\cdot i+1, i+1)$
    
    $$x = i^2 \land i < n \rightarrow x + 2\cdot i + 1 = (i + 1)^2$$

  - $I(x, n) \rightarrow Q(x)$
    
    $$x = n^2 \rightarrow x = n^2$$

The computation of a square as a sum of odd numbers.
Example

▶ Program

\[
\text{for}(i=0; \ i<n; \ i++) \ x = x + \frac{1}{2^i};
\]

▶ Precondition \( P(x) : \iff x = 0 \)

<table>
<thead>
<tr>
<th>( x )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i )</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4=n</td>
</tr>
</tbody>
</table>

▶ Postcondition \( Q(x) : \iff x + \frac{1}{2^{n-1}} = 2 \)

▶ Loop invariant \( I(x, i) : \iff x + \frac{1}{2^{i-1}} = 2 \)

▶ \( P(x) \rightarrow I(x, 0) \)

\[
x = 0 \rightarrow x + \frac{1}{2^{0-1}} = 2
\]

▶ \( I(x, i) \land i < n \rightarrow I(x + \frac{1}{2^i}, i + 1) \)

\[
x + \frac{1}{2^{i-1}} = 2 \land i < n \rightarrow x + \frac{1}{2^i} + \frac{1}{2^i} = 2
\]

▶ \( I(x, n) \rightarrow Q(x) \)

\[
x + \frac{1}{2^{n-1}} = 2 \rightarrow x + \frac{1}{2^{n-1}} = 2
\]

The approximation of a value by a convergent series.
Example

- **Program**

  \[
  \text{for}(i=0; i<n; i++) \ x = x + a(i);
  \]

- **Precondition** \( P(x) : \iff x = 0 \)

  \[
  \begin{array}{c|cccc}
  x & 0 & 2 & 5 & 10 & 17 \\
  i & 0 & 1 & 2 & 3 & 4=n \\
  \end{array}
  \quad a = [2, 3, 5, 7]
  \]

- **Postcondition** \( Q(x) : \iff x = \sum_{j=0}^{n-1} a(j) \)

- **Loop invariant** \( I(x, i) : \iff x = \sum_{j=0}^{i-1} a(j) \)

  - \( P(x) \rightarrow I(x, 0) \)

    \[
    x = 0 \rightarrow x = \sum_{j=0}^{i-1} a(j)
    \]

  - \( I(x, i) \land i < n \rightarrow I(x + a(j), i + 1) \)

    \[
    x = \sum_{j=0}^{i-1} a(j) \land i < n \rightarrow x + a(i) = \sum_{j=0}^{i} a(j)
    \]

  - \( I(x, n) \rightarrow Q(x) \)

    \[
    x = \sum_{j=0}^{n-1} a(j) \rightarrow x = \sum_{j=0}^{n-1} a(j)
    \]

The summation of an array of values.