Application of Mathematical Logic in Functional Program Verification

Nikolaj Popov and Tudor Jebelean

Research Institute for Symbolic Computation, Linz

{popov,jebelean}@risc.uni-linz.ac.at

Outline

Functional Program Verification
Total Correctness
Building up Correct Programs
Coherent Programs. Recursion
Soundness and Completeness
Double (Multiple) Recursion Program Scheme. Termination

Conclusion and Discussions

Preconditions and Postconditions. Total Correctness

Given the triple

$\{I\}F\{O\}$ (Input condition, Function definition, Output condition)

Total Correctness Formula

$(\forall n : I[n])\ (F[n]\downarrow \wedge O[n, F[n]])$

Example

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$

$pow[x, n] =$ **If** $n = 0$ **then** 1 **else** $x * pow[x, n-1]$

$\{x^n = pow[x, n]\}$

$(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (pow[x, n]\downarrow \wedge\ x^n = pow[x, n])$

Preconditions and Postconditions. Total Correctness

Given the triple

$\{I\}F\{O\}$ (Input condition, Function definition, Output condition)

Total Correctness Formula

$(\forall n : I[n])\ (F[n] \downarrow \wedge O[n, F[n]])$

Example

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$

$pow[x, n] =$ **If** $n = 0$ **then** 1 **else** $x * pow[x, n-1]$

$\{x^n = pow[x, n]\}$

$(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (pow[x, n] \downarrow \wedge\ x^n = pow[x, n])$

Preconditions and Postconditions. Total Correctness

Given the triple

$\{I\}F\{O\}$ (Input condition, Function definition, Output condition)

Total Correctness Formula

$(\forall n : I[n])\ (F[n] \downarrow \wedge O[n, F[n]])$

Example

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$

$pow[x, n] =$ **If** $n = 0$ **then** 1 **else** $x * pow[x, n-1]$

$\{x^n = pow[x, n]\}$

$(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (pow[x, n] \downarrow \wedge\ x^n = pow[x, n])$

Building up Correct Programs

Basic Functions e.g. +, -, *, etc.

New Functions in Terms of Already Known Functions

- Input and output predicates;
- Prove total correctness.

Modularity. After proving correctness, use only the specification.

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$ *Input condition*

$pow[x, n] = \ldots$

$\{x^n = pow[x, n]\}$ *Output condition*

Building up Correct Programs

Basic Functions e.g. +, -, *, etc.

New Functions in Terms of Already Known Functions

- Input and output predicates;
- Prove total correctness;

Modularity. After proving correctness, use only the specification.

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$ *Input condition*

$pow[x, n] = \ldots$

$\{x^n = pow[x, n]\}$ *Output condition*

Building up Correct Programs

Basic Functions e.g. +, -, *, etc.

New Functions in Terms of Already Known Functions

- Input and output predicates;
- Prove total correctness;

Modularity. After proving correctness, use only the specification.

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$ *Input condition*

$pow[x, n] = \ldots$

$\{x^n = pow[x, n]\}$ *Output condition*

Building up Correct Programs

Basic Functions e.g. +, -, *, etc.

New Functions in Terms of Already Known Functions

- Input and output predicates;
- Prove total correctness;

Modularity. After proving correctness, use only the specification.

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$ *Input condition*

$pow[x, n] = \ldots$

$\{x^n = pow[x, n]\}$ *Output condition*

Building up Correct Programs

Basic Functions e.g. +, -, *, etc.

New Functions in Terms of Already Known Functions

- Input and output predicates;
- Prove total correctness;

Modularity. After proving correctness, use only the specification.

$\{x \in \mathbb{R} \wedge n \in \mathbb{N}\}$ *Input condition*

$pow[x, n] = \ldots$

$\{x^n = pow[x, n]\}$ *Output condition*

Building up Correct Programs

Appropriate values for the auxiliary functions

No input condition of an auxiliary function will be violated

Example

$F[x] =$ **If** $Q[x]$ **then** $H[x]$ **else** $G[x]$

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_H[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_G[x])$

Building up Correct Programs

Appropriate values for the auxiliary functions

No input condition of an auxiliary function will be violated

Example

$F[x] =$ **If** $Q[x]$ **then** $H[x]$ **else** $G[x]$

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_H[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_G[x])$

Building up Correct Programs

Appropriate values for the auxiliary functions

No input condition of an auxiliary function will be violated

Example

$F[x] =$ **If** $Q[x]$ **then** $H[x]$ **else** $G[x]$

- $(\forall x: I_F[x])\ (Q[x] \Rightarrow I_H[x])$
- $(\forall x: I_F[x])\ (\neg Q[x] \Rightarrow I_G[x])$

Building up Correct Programs

Appropriate values for the auxiliary functions

No input condition of an auxiliary function will be violated

Example

$F[x] =$ **If** $Q[x]$ **then** $H[x]$ **else** $G[x]$

- $(\forall x: \ I_F[x]) \ (Q[x] \Rightarrow I_H[x])$
- $(\forall x: \ I_F[x]) \ (\neg Q[x] \Rightarrow I_G[x])$

Building up Correct Programs

Appropriate values for the auxiliary functions

No input condition of an auxiliary function will be violated

Example

$F[x] =$ **If** $Q[x]$ **then** $H[x]$ **else** $G[x]$

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_H[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_G[x])$

Coherent Programs

Simple Recursive Programs

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

Conditions for coherency

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_S[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_F[R[x]])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_R[x])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow I_C[x, y])$

Coherent Programs

Simple Recursive Programs

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

Conditions for coherency

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_S[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_F[R[x]])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_R[x])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow I_C[x, y])$

Coherent Programs

Simple Recursive Programs

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

Conditions for coherency

- $(\forall x : \; I_F[x]) \; (Q[x] \Rightarrow I_S[x])$
- $(\forall x : \; I_F[x]) \; (\neg Q[x] \Rightarrow I_F[R[x]])$
- $(\forall x : \; I_F[x]) \; (\neg Q[x] \Rightarrow I_R[x])$
- $(\forall x, y : \; I_F[x]) \; (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow I_C[x, y])$

Coherent Programs

Simple Recursive Programs

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

Conditions for coherency

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_S[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_F[R[x]])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_R[x])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow I_C[x, y])$

Coherent Programs

Simple Recursive Programs

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

Conditions for coherency

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_S[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_F[R[x]])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_R[x])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow I_C[x, y])$

Coherent Programs

Simple Recursive Programs

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

Conditions for coherency

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_S[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_F[R[x]])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_R[x])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow I_C[x, y])$

Verification Conditions Generation

Simple Recursive Program

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

is correct if the verification conditions hold

- $(\forall x : I_F[x]) \; (Q[x] \Rightarrow O_F[x, S[x]])$
- $(\forall x, y : I_F[x]) \; (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow O_F[x, C[x, y]])$
- $(\forall x : I_F[x]) \; (F'[x] = 0)$
- where:

$$F'[x] = \textbf{If } Q[x] \textbf{ then } 0 \textbf{ else } F'[R[x]]$$

Verification Conditions Generation

Simple Recursive Program

$F[x] = $ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

is correct if the verification conditions hold

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow O_F[x, S[x]])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow O_F[x, C[x, y]])$
- $(\forall x : I_F[x])\ (F'[x] = 0)$
- where:

$F'[x] = $ **If** $Q[x]$ **then** 0 **else** $F'[R[x]]$

Verification Conditions Generation

Simple Recursive Program

$F[x] = \textbf{ If } Q[x] \textbf{ then } S[x] \textbf{ else } C[x, F[R[x]]]$

is correct if the verification conditions hold

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow O_F[x, S[x]])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow O_F[x, C[x, y]])$
- $(\forall x : I_F[x])\ (F'[x] = 0)$
- where:

$$F'[x] = \textbf{ If } Q[x] \textbf{ then } 0 \textbf{ else } F'[R[x]]$$

Verification Conditions Generation

Simple Recursive Program

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

is correct if the verification conditions hold

- $(\forall x : I_F[x]) \ (Q[x] \Rightarrow O_F[x, S[x]])$
- $(\forall x, y : I_F[x]) \ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow O_F[x, C[x, y]])$
- $(\forall x : I_F[x]) \ (F'[x] = 0)$
- where:

$$F'[x] = \textbf{ If } Q[x] \textbf{ then } 0 \textbf{ else } F'[R[x]]$$

Verification Conditions Generation

Simple Recursive Program

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R[x]]]$

is correct if the verification conditions hold

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow O_F[x, S[x]])$
- $(\forall x, y : I_F[x])\ (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow O_F[x, C[x, y]])$
- $(\forall x : I_F[x])\ (F'[x] = 0)$
- where:

$$F'[x] = \textbf{If } Q[x] \textbf{ then } 0 \textbf{ else } F'[R[x]]$$

Verification Conditions Generation

Simple Recursive Program

$F[x] = \textbf{ If } Q[x] \textbf{ then } S[x] \textbf{ else } C[x, F[R[x]]]$

is correct if the verification conditions hold

- $(\forall x : \; I_F[x]) \; (Q[x] \Rightarrow O_F[x, S[x]])$
- $(\forall x, y : \; I_F[x]) \; (\neg Q[x] \wedge O_F[R[x], y] \Rightarrow O_F[x, C[x, y]])$
- $(\forall x : \; I_F[x]) \; (F'[x] = 0)$
- where:

$$F'[x] = \textbf{ If } Q[x] \textbf{ then } 0 \textbf{ else } F'[R[x]]$$

Soundness and Completeness

$\langle Program, Specification \rangle \xrightarrow{VCG} VerificationConditions$

$\langle F[x], \langle I_F[x], O_F[x, F[x]] \rangle \rangle \xrightarrow{VCG} VerificationConditions$

Soundness

if $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

then $\forall x \, (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

Completeness

if $\forall x \, (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

then $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

Soundness and Completeness

$$\langle Program, Specification \rangle \xrightarrow{VCG} VerificationConditions$$

$$\langle F[x], \langle I_F[x], O_F[x, F[x]] \rangle \rangle \xrightarrow{VCG} VerificationConditions$$

Soundness

if $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

then $\forall x\, (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

Completeness

if $\forall x\, (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

then $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

Soundness and Completeness

$\langle Program, Specification\rangle \overset{VCG}{\longrightarrow} VerificationConditions$

$\langle F[x], \langle I_F[x], O_F[x, F[x]]\rangle\rangle \overset{VCG}{\longrightarrow} VerificationConditions$

Soundness

if $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

then $\forall x\ (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

Completeness

if $\forall x\ (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

then $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

Soundness and Completeness

$$\langle \mathit{Program}, \mathit{Specification} \rangle \xrightarrow{\mathit{VCG}} \mathit{VerificationConditions}$$

$$\langle F[x], \langle I_F[x], O_F[x, F[x]] \rangle \rangle \xrightarrow{\mathit{VCG}} \mathit{VerificationConditions}$$

Soundness

if $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

then $\forall x \, (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

Completeness

if $\forall x \, (I[x] \Rightarrow F[x] \downarrow \wedge O[x, F[x]])$

then $\models \varphi_1 \wedge \cdots \wedge \varphi_n$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \quad \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

is coherent if

- $(\forall n : \mathbb{N})\ (n \neq 0 \Rightarrow n \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n = 0 \Rightarrow \mathbb{D})$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

is coherent if

- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow n \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow \mathbf{T})$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \quad \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

is coherent if

- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow n \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow \mathbf{T})$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \begin{array}{l} \textbf{If } n = 0 \textbf{ then } 0 \\ \textbf{else } n + Sum[n-1]. \end{array}$$

is coherent if

- $(\forall n : \mathbb{N})\ (n \neq 0 \Rightarrow n \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n = 0 \Rightarrow \mathbf{T})$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \quad \textbf{If } n = 0 \textbf{ then } 0 \\ \textbf{else } n + Sum[n-1].$$

is correct if and only if

- $(\forall n : \mathbb{N})\ (n = 0 \Rightarrow 0 = \frac{n(n+1)}{2})$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = \frac{(n-1)((n-1)+1)}{2} \Rightarrow n + m = \frac{n(n+1)}{2})$
- $(\forall n : \mathbb{N})\ (Sum'[n] = 0)$
- where:

$$Sum'[n] = \textbf{ If } n = 0 \textbf{ then } 0 \textbf{ else } Sum'[n-1]$$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \begin{array}{l}\textbf{If } n = 0 \textbf{ then } 0\\ \textbf{else } n + Sum[n-1].\end{array}$$

is correct if and only if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow 0 = \frac{n(n+1)}{2})$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = \frac{(n-1)((n-1)+1)}{2} \ \Rightarrow n + m = \frac{n(n+1)}{2})$
- $(\forall n : \mathbb{N})\ (Sum'[n] = 0)$
- where:

$$Sum'[n] = \textbf{If } n = 0 \textbf{ then } 0 \textbf{ else } Sum'[n-1]$$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \quad \textbf{If } n = 0 \textbf{ then } 0$$
$$\textbf{else } n + Sum[n-1].$$

is correct if and only if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow 0 = \frac{n(n+1)}{2})$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = \frac{(n-1)((n-1)+1)}{2} \ \Rightarrow n + m = \frac{n(n+1)}{2})$
- $(\forall n : \mathbb{N})\ (Sum'[n] = 0)$
- where:

$$Sum'[n] = \ \textbf{If } n = 0 \textbf{ then } 0 \textbf{ else } Sum'[n-1]$$

Example

Sum $(\forall n:\mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \begin{array}{l}\textbf{If } n = 0 \textbf{ then } 0 \\ \textbf{else } n + Sum[n-1].\end{array}$$

is correct if and only if

- $(\forall n:\mathbb{N})\ (n = 0 \ \Rightarrow 0 = \frac{n(n+1)}{2})$
- $(\forall n, m:\mathbb{N})\ (n \neq 0 \wedge m = \frac{(n-1)((n-1)+1)}{2} \ \Rightarrow n + m = \frac{n(n+1)}{2})$
- $(\forall n:\mathbb{N})\ (Sum'[n] = 0)$
- where:

$$Sum'[n] = \textbf{ If } n = 0 \textbf{ then } 0 \textbf{ else } Sum'[n-1]$$

Example

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \textbf{If } n = 0 \textbf{ then } 0 \textbf{ else } n + Sum[n-1].$$

is correct if and only if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow 0 = \frac{n(n+1)}{2})$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = \frac{(n-1)((n-1)+1)}{2} \ \Rightarrow n + m = \frac{n(n+1)}{2})$
- $(\forall n : \mathbb{N})\ (Sum'[n] = 0)$
- where:

$$Sum'[n] = \textbf{If } n = 0 \textbf{ then } 0 \textbf{ else } Sum'[n-1]$$

Example

Binary powering $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ P[x, n] = x^n$

$$P[x,n] = \begin{array}{l} \textbf{If } n = 0 \textbf{ then } 1 \\ \textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x * x, n/2] \\ \textbf{else } x * P[x * x, (n-1)/2]. \end{array}$$

is coherent if and only if

- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \ldots \Rightarrow \mathbb{T})$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \Rightarrow \mathrm{Even}[n])$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \Rightarrow \mathrm{Even}[n-1])$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \Rightarrow n/2 \in \mathbb{N})$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \Rightarrow (n-1)/2 \in \mathbb{N})$

Example

Binary powering $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ P[x, n] = x^n$

$$P[x,n] = \begin{array}{l} \textbf{If } n = 0 \textbf{ then } 1 \\ \textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x * x, n/2] \\ \textbf{else } x * P[x * x, (n-1)/2]. \end{array}$$

is coherent if and only if

- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \ \ldots \ \Rightarrow \mathbb{T})$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \ \Rightarrow \mathrm{Even}[n])$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \ \Rightarrow \mathrm{Even}[n-1])$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \ \Rightarrow n/2 \in \mathbb{N})$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \ \Rightarrow (n-1)/2 \in \mathbb{N})$

Example

Binary powering $(\forall x:\mathbb{R})(\forall n:\mathbb{N})\ P[x,n]=x^n$

$$P[x,n] = \begin{array}{l} \textbf{If } n=0 \textbf{ then } 1 \\ \textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x*x,n/2] \\ \textbf{else } x*P[x*x,(n-1)/2]. \end{array}$$

is correct if and only if

- $(\forall x:\mathbb{R})(\forall n:\mathbb{N})\ (n=0\ \Rightarrow 1=x^n)$
- $(\forall x,m:\mathbb{R})(\forall n:\mathbb{N})\ (n\neq 0\wedge \mathrm{Even}[n]\wedge m=(x*x)^{n/2}\ \Rightarrow m=x^n)$
- $(\forall x,m:\mathbb{R})(\forall n:\mathbb{N})\ (n\neq 0\wedge \neg\mathrm{Even}[n]\wedge m=(x*x)^{(n-1)/2}\ \Rightarrow x*m=x^n)$
- $(\forall x:\mathbb{R})(\forall n:\mathbb{N})\ (P'[x,n]=\mathbb{T})$

Example

Binary powering $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ P[x, n] = x^n$

$$P[x,n] = \begin{array}{l} \textbf{If } n = 0 \textbf{ then } 1 \\ \textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x * x, n/2] \\ \textbf{else } x * P[x * x, (n-1)/2]. \end{array}$$

is correct if and only if

- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n = 0\ \Rightarrow 1 = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \wedge m = (x * x)^{n/2}\ \Rightarrow m = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \wedge m = (x * x)^{(n-1)/2}\ \Rightarrow x * m = x^n)$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (P'[x, n] = \mathbb{T})$

Counter-Example

Binary powering $(\forall x:\mathbb{R})(\forall n:\mathbb{N})\ P[x,n]=x^n$

$$P[x,n] = \begin{array}{l}\textbf{If } n=0 \textbf{ then } 0\\ \textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x*x,n/2]\\ \textbf{else } x*P[x*x,(n-1)/2].\end{array}$$

is correct if and only if

- $(\forall x:\mathbb{R})(\forall n:\mathbb{N})\ (n=0\ \Rightarrow 0=x^n)$
- $(\forall x,m:\mathbb{R})(\forall n:\mathbb{N})\ (n\neq 0\wedge \mathrm{Even}[n]\wedge m=(x*x)^{n/2}\ \Rightarrow m=x^n)$
- $(\forall x,m:\mathbb{R})(\forall n:\mathbb{N})\ (n\neq 0\wedge \neg\mathrm{Even}[n]\wedge m=(x*x)^{(n-1)/2}\ \Rightarrow x*m=x^n)$
- $(\forall x:\mathbb{R})(\forall n:\mathbb{N})\ (P'[x,n]=\mathbb{T})$

Counter-Example

Binary powering $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ P[x, n] = x^n$

$$P[x,n] = \begin{array}{l} \textbf{If } n = 0 \textbf{ then } \mathbf{0} \\ \textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x*x, n/2] \\ \textbf{else } x * P[x*x, (n-1)/2]. \end{array}$$

is correct if and only if

- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow \mathbf{0} = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \wedge m = (x*x)^{n/2} \ \Rightarrow m = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \wedge m = (x*x)^{(n-1)/2} \ \Rightarrow x * m = x^n)$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (P'[x, n] = \mathbb{T})$

Counter-Example

Binary powering $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ P[x, n] = x^n$

$$P[x, n] = \textbf{If } n = 0 \textbf{ then } 1$$
$$\textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x, n/2] \qquad \%\ \textit{but not } x * x$$
$$\textbf{else } x * P[x * x, (n - 1)/2].$$

is correct if and only if

- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n = 0 \Rightarrow 1 = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \wedge m = (x)^{n/2} \Rightarrow m = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \wedge m = (x * x)^{(n-1)/2} \Rightarrow x * m = x^n)$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (P'[x, n] = \mathbb{T})$

Counter-Example

Binary powering $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ P[x, n] = x^n$

$$P[x,n] = \quad \textbf{If } n = 0 \textbf{ then } 1$$
$$\textbf{elseif } \mathrm{Even}[n] \textbf{ then } P[x, n/2] \qquad \%\ \textit{but not } x * x$$
$$\textbf{else } x * P[x * x, (n-1)/2].$$

is correct if and only if

- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow 1 = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \mathrm{Even}[n] \wedge m = (x)^{n/2} \ \Rightarrow m = x^n)$
- $(\forall x, m : \mathbb{R})(\forall n : \mathbb{N})\ (n \neq 0 \wedge \neg\mathrm{Even}[n] \wedge m = (x * x)^{(n-1)/2} \ \Rightarrow x * m = x^n)$
- $(\forall x : \mathbb{R})(\forall n : \mathbb{N})\ (P'[x, n] = \mathbb{T})$

Coherent Recursive Programs

Double (Multiple) Recursion Program Scheme

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R_1[x]], F[R_2[x]]]$

Conditions for coherence

- $(\forall x : I_F[x])\ (Q[x] \Rightarrow I_S[x])$

- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_F[R_1[x]])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_F[R_2[x]])$

- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_{R_1}[x])$
- $(\forall x : I_F[x])\ (\neg Q[x] \Rightarrow I_{R_2}[x])$

- $(\forall x, y, z : I_F[x])\ (\neg Q[x] \wedge O_F[R_1[x], y] \wedge O_F[R_2[x], z] \Rightarrow I_C[x, y, z])$

Coherent Recursive Programs

Double (Multiple) Recursion Program Scheme

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R_1[x]], F[R_2[x]]]$

Conditions for coherence

- $(\forall x : I_F[x]) \ (Q[x] \Rightarrow I_S[x])$

- $(\forall x : I_F[x]) \ (\neg Q[x] \Rightarrow I_F[R_1[x]])$
- $(\forall x : I_F[x]) \ (\neg Q[x] \Rightarrow I_F[R_2[x]])$

- $(\forall x : I_F[x]) \ (\neg Q[x] \Rightarrow I_{R_1}[x])$
- $(\forall x : I_F[x]) \ (\neg Q[x] \Rightarrow I_{R_2}[x])$

- $(\forall x, y, z : I_F[x]) \ (\neg Q[x] \wedge O_F[R_1[x], y] \wedge O_F[R_2[x], z] \Rightarrow I_C[x, y, z])$

Coherent Recursive Programs

Double (Multiple) Recursion Program Scheme

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R_1[x]], F[R_2[x]]]$

Conditions for Partial Correctness

- $(\forall x : \; I_F[x]) \; (Q[x] \Rightarrow O_F[x, S[x]])$
- $(\forall x, y, z : \; I_F[x]) \; (\neg Q[x] \wedge O_F[R_1[x], y] \wedge O_F[R_2[x], z] \Rightarrow O_F[x, C[x, y, z]])$

Coherent Recursive Programs

Double (Multiple) Recursion Program Scheme

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R_1[x]], F[R_2[x]]]$

Condition for Termination

- $(\forall x : \; I_F[x]) \; (F'[x] = \mathbf{T})$
- where:

$$F'[x] = \textbf{ If } Q[x] \textbf{ then } \mathbf{T} \textbf{ else } F'[R_1[x]] \wedge F'[R_2[x]]$$

Coherent Recursive Programs

Double (Multiple) Recursion Program Scheme

$F[x] =$ **If** $Q[x]$ **then** $S[x]$ **else** $C[x, F[R_1[x]], F[R_2[x]]]$

Condition for Termination

- $(\forall x : \; I_F[x]) \; (F'[x] = \mathbf{T})$
- where:

$$F'[x] = \textbf{ If } Q[x] \textbf{ then T else } F'[R_1[x]] \wedge F'[R_2[x]]$$

Example Factorial

Fact $(\forall n : \mathbb{N})\ (Fact[n] = n!)$

$$Fact[n] = \textbf{If } n = 0 \textbf{ then } 1 \quad \textbf{else } n * Fact[n-1].$$

is coherent if

- $(\forall n : \mathbb{N})\ (n = 0 \Rightarrow \mathbf{T})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \Rightarrow n - 1 \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \Rightarrow \mathbf{T})$

Example Factorial

Fact $(\forall n : \mathbb{N})\ (Fact[n] = n!)$

$$Fact[n] = \quad \textbf{If } n = 0 \textbf{ then } 1 \quad \textbf{else } n * Fact[n-1].$$

is coherent if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow \mathbf{T})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow n - 1 \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow \mathbf{T})$

Example Factorial

Fact $(\forall n : \mathbb{N})\ (Fact[n] = n!)$

$$Fact[n] = \textbf{If } n = 0 \textbf{ then } 1 \quad \textbf{else } n * Fact[n-1].$$

is partially correct if

- $(\forall n : \mathbb{N})\ (n = 0 \Rightarrow 1 = n!)$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = (n-1)! \Rightarrow n * m = n!)$

Example Factorial

Fact $(\forall n : \mathbb{N})\ (Fact[n] = n!)$

$$Fact[n] = \quad \textbf{If } n = 0 \textbf{ then } 1$$
$$\textbf{else } n * Fact[n-1].$$

is partially correct if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow 1 = n!)$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = (n-1)! \ \Rightarrow n * m = n!)$

Example Factorial

Fact $(\forall n : \mathbb{N})\ (Fact[n] = n!)$

$$Fact[n] = \textbf{If } n = 0 \textbf{ then } 1 \\ \textbf{else } n * Fact[n-1].$$

terminates if

- $(\forall n : \mathbb{N})\ (Fact'[n] = \mathbf{T})$
- where:

$$Fact'[n] = \textbf{If } n = 0 \textbf{ then } \mathbf{T} \\ \textbf{else } Fact'[n-1].$$

Example Factorial

Fact $(\forall n : \mathbb{N})\ (Fact[n] = n!)$

$$Fact[n] = \textbf{If } n = 0 \textbf{ then } 1$$
$$\textbf{else } n * Fact[n-1].$$

terminates if

- $(\forall n : \mathbb{N})\ (Fact'[n] = \mathbf{T})$
- where:

$$Fact'[n] = \textbf{If } n = 0 \textbf{ then } \mathbf{T}$$
$$\textbf{else } Fact'[n-1].$$

Example Sum

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \quad \textbf{If } n = 0 \textbf{ then } 0$$
$$\textbf{else } n + Sum[n-1].$$

is coherent if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow \mathbf{T})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow n - 1 \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow \mathbf{T})$

Example Sum

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

is coherent if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow \mathbf{T})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow n - 1 \in \mathbb{N})$
- $(\forall n : \mathbb{N})\ (n \neq 0 \ \Rightarrow \mathbf{T})$

Example Sum

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

is partially correct if

- $(\forall n : \mathbb{N})\ (n = 0 \Rightarrow 0 = \frac{n(n+1)}{2})$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = \frac{(n-1)((n-1)+1)}{2} \Rightarrow n + m = \frac{n(n+1)}{2})$

Example Sum

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

is partially correct if

- $(\forall n : \mathbb{N})\ (n = 0 \ \Rightarrow 0 = \frac{n(n+1)}{2})$
- $(\forall n, m : \mathbb{N})\ (n \neq 0 \wedge m = \frac{(n-1)((n-1)+1)}{2} \ \Rightarrow n + m = \frac{n(n+1)}{2})$

Example Sum

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \quad \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

terminates if

- $(\forall n : \mathbb{N})\ (Sum'[n] = \mathbf{T})$
- where:

$$Sum'[n] = \quad \textbf{If } n = 0 \textbf{ then } \mathbf{T} \quad \textbf{else } Sum'[n-1].$$

Example Sum

Sum $(\forall n : \mathbb{N})\ (Sum[n] = \frac{n(n+1)}{2})$

$$Sum[n] = \textbf{If } n = 0 \textbf{ then } 0 \quad \textbf{else } n + Sum[n-1].$$

terminates if

- $(\forall n : \mathbb{N})\ (Sum'[n] = \mathbf{T})$
- where:

$$Sum'[n] = \textbf{If } n = 0 \textbf{ then } \mathbf{T} \quad \textbf{else } Sum'[n-1].$$

Neville's Algorithm

Specification

Given a field K, two non-empty tuples x, a over K of same length n, s.t. $(\forall i, j)(i, j = 1, \ldots, n \wedge i \neq j \Rightarrow x_i \neq x_j)$

Find a polynomial $p \in \mathcal{P}[K]$, s.t. $deg[p] \leq n - 1$ and $(\forall i)(i = 1, \ldots, n \Rightarrow Eval[p, x_i] = a_i)$

Algorithm

$$p[x, a] = \textbf{If } \|a\| \leq 1 \textbf{ then } First[a]$$

$$\textbf{else } \frac{(\mathcal{X} - First[x])(p[Tail[x], Tail[a]]) - (\mathcal{X} - Last[x])(p[Bgn[x], Bgn[a]])}{Last[x] - First[x]}.$$

Neville's Algorithm

Specification

Given a field K, two non-empty tuples x, a over K of same length n, s.t. $(\forall i, j)(i, j = 1, \ldots, n \wedge i \neq j \Rightarrow x_i \neq x_j)$

Find a polynomial $p \in \mathcal{P}[K]$, s.t. $deg[p] \leq n - 1$ and $(\forall i)(i = 1, \ldots, n \Rightarrow Eval[p, x_i] = a_i)$

Algorithm

$$p[x, a] = \textbf{ If } \|a\| \leq 1 \textbf{ then } First[a]$$

$$\textbf{else } \frac{(\mathcal{X} - First[x])(p[Tail[x], Tail[a]]) - (\mathcal{X} - Last[x])(p[Bgn[x], Bgn[a]])}{Last[x] - First[x]}.$$

Neville's Algorithm

is coherent if

- $(\forall x, a)(IsTuple[a] \wedge IsTuple[x] \wedge \|a\| \geq 1 \wedge$

  $(\forall i, j)(i, j = 1 \ldots \|a\| \wedge i \neq j \Rightarrow x_i \neq x_j) \wedge \|a\| \leq 1 \ \Rightarrow IsTuple[a] \wedge \|a\| \geq 1)$
- $(\forall x, a)(IsTuple[a] \wedge IsTuple[x] \wedge \|a\| \geq 1 \wedge$

  $(\forall i, j)(i, j = 1 \ldots \|a\| \wedge i \neq j \Rightarrow x_i \neq x_j) \wedge \neg(\|a\| \leq 1) \ \Rightarrow$

  $IsTuple[Tail[x]] \wedge IsTuple[Tail[a]] \wedge \|Tail[a]\| = \|Tail[x]\| \wedge \|Tail[a]\| \geq 1$

  $\wedge(\forall i, j)(i, j = 1 \ldots \|Tail[a]\| \wedge i \neq j \Rightarrow Tail[x]_i \neq Tail[x]_j)$
- ...
- ...

Neville's Algorithm

is coherent if

- $(\forall x, a)(IsTuple[a] \wedge IsTuple[x] \wedge \|a\| \geq 1 \wedge$

  $(\forall i, j)(i, j = 1 \ldots \|a\| \wedge i \neq j \Rightarrow x_i \neq x_j) \wedge \|a\| \leq 1 \ \Rightarrow IsTuple[a] \wedge \|a\| \geq 1)$
- $(\forall x, a)(IsTuple[a] \wedge IsTuple[x] \wedge \|a\| \geq 1 \wedge$

  $$(\forall i, j)(i, j = 1 \ldots \|a\| \wedge i \neq j \Rightarrow x_i \neq x_j) \wedge \neg(\|a\| \leq 1) \ \Rightarrow$$

  $IsTuple[Tail[x]] \wedge IsTuple[Tail[a]] \wedge \|Tail[a]\| = \|Tail[x]\| \wedge \|Tail[a]\| \geq 1$

  $$\wedge(\forall i, j)(i, j = 1 \ldots \|Tail[a]\| \wedge i \neq j \Rightarrow Tail[x]_i \neq Tail[x]_j)$$
- ...
- ...

Neville's Algorithm

is partially correct if and only if

- $\cdots \wedge IsPoly[p_1] \wedge IsPoly[p_2] \Rightarrow IsPoly[\frac{(\mathcal{X}-First[x])p_1-(\mathcal{X}-Last[x])p_2}{Last[x]-First[x]}]$
- $\cdots \wedge (\forall i)(i = 1 \ldots \|Tail[x]\|)(Eval[p_1, Tail[x]_i]) = Tail[a]_i$

  $$\wedge(\forall i)(i = 1 \ldots \|Bgn[x]\|)(Eval[p_2, Bgn[x]_i]) = Tail[a]_i$$

  $$\Rightarrow (\forall i)(i = 1 \ldots \|a\|)(Eval[\frac{(\mathcal{X} - First[x])p_1 - (\mathcal{X} - Last[x])p_2}{Last[x] - First[x]}, x_i] = a_i)$$

- $\cdots \wedge deg[p_1] \leq \|Tail[a]\| - 1 \wedge deg[p_2] \leq \|Bgn[a]\| - 1$

  $$\Rightarrow deg[\frac{(\mathcal{X} - First[x])p_1 - (\mathcal{X} - Last[x])p_2}{Last[x] - First[x]} \leq \|a\| - 1$$

- ...

Neville's Algorithm

is partially correct if and only if

- $\cdots \wedge IsPoly[p_1] \wedge IsPoly[p_2] \Rightarrow IsPoly[\frac{(\mathcal{X}-First[x])p_1-(\mathcal{X}-Last[x])p_2}{Last[x]-First[x]}]$

- $\cdots \wedge (\forall i)(i = 1 \ldots \|Tail[x]\|)(Eval[p_1, Tail[x]_i]) = Tail[a]_i$

  $\wedge(\forall i)(i = 1 \ldots \|Bgn[x]\|)(Eval[p_2, Bgn[x]_i]) = Tail[a]_i$

  $\Rightarrow (\forall i)(i = 1 \ldots \|a\|)(Eval[\frac{(\mathcal{X} - First[x])p_1 - (\mathcal{X} - Last[x])p_2}{Last[x] - First[x]}, x_i] = a_i)$

- $\cdots \wedge deg[p_1] \leq \|Tail[a]\| - 1 \wedge deg[p_2] \leq \|Bgn[a]\| - 1$

  $\Rightarrow deg[\frac{(\mathcal{X} - First[x])p_1 - (\mathcal{X} - Last[x])p_2}{Last[x] - First[x]} \leq \|a\| - 1$

- ...

Neville's Algorithm

terminates if and only if

- $(\forall x, a : \mathit{IsTuple}[a] \wedge \mathit{IsTuple}[x] \wedge \|a\| = \|x\|) \quad p'[x, a] = \mathbf{T}$
- Where:

$$p'[x, a] = \quad \textbf{If } \|a\| \leq 1 \textbf{ then T} \quad \textbf{else } p'[\mathit{Tail}[x], \mathit{Tail}[a]] \wedge p'[\mathit{Bgn}[x], \mathit{Bgn}[a]].$$

Neville's Algorithm

terminates if and only if

- $(\forall x, a : \mathit{IsTuple}[a] \wedge \mathit{IsTuple}[x] \wedge \|a\| = \|x\|) \quad p'[x, a] = \mathbf{T}$
- Where:

$$p'[x, a] = \quad \textbf{If } \|a\| \leq 1 \textbf{ then T}$$
$$\textbf{else } p'[\mathit{Tail}[x], \mathit{Tail}[a]] \wedge p'[\mathit{Bgn}[x], \mathit{Bgn}[a]].$$

Neville's Algorithm

terminates if and only if

- $(\forall x, a : \mathit{IsTuple}[a] \wedge \mathit{IsTuple}[x] \wedge \|a\| = \|x\|) \quad p'[x, a] = \mathbf{T}$
- Where:

$$p'[x, a] = \quad \textbf{If } \|a\| \leq 1 \textbf{ then T}$$
$$\textbf{else } p'[\mathit{Tail}[x], \mathit{Tail}[a]] \wedge p'[\mathit{Bgn}[x], \mathit{Bgn}[a]].$$

Outline

Functional Program Verification
Total Correctness
Building up Correct Programs
Coherent Programs. Recursion
Soundness and Completeness
Double (Multiple) Recursion Program Scheme. Termination

Conclusion and Discussions

Conclusions and Discussion

- The problem of proving program correctness is translated into a problem of proving first order formulae;
- Prove by hand;
- Prove by an automatic theorem prover.

Conclusions and Discussion

- The problem of proving program correctness is translated into a problem of proving first order formulae;
- Prove by hand;
- Prove by an automatic theorem prover.

Conclusions and Discussion

- The problem of proving program correctness is translated into a problem of proving first order formulae;
- Prove by hand;
- Prove by an automatic theorem prover.