CAS 701 Fall 2002

06 Recursive Definition and Inductive Proof

Instructor: W. M. Farmer

Revised: 30 November 2002

What is Recursion?

- **Recursion** is a method of defining a structure or operation in terms of itself
  - One of the most fundamental ideas of computing
  - Can make some specifications, descriptions, and programs easier to express and prove correct

- **Induction** is a method of proof based on a recursively defined structure
  - The recursively defined structure and the proof method are specified by an **induction principle**

- The terms "recursion" and "induction" are often used interchangeably

Example: Natural Numbers

- Recursive definition:
  1. $0 \in \mathbf{N}$
  2. If $n \in \mathbf{N}$, then $S(n) \in \mathbf{N}$
  3. The members of $\mathbf{N}$ are distinct ("no confusion")
  4. $\mathbf{N}$ is the smallest such set ("no junk")

- Induction principle:

$$
\begin{array}{l}
\forall P : \mathbf{N} \rightarrow * \, . \\
\quad [P(0) \wedge (\forall x : \mathbf{N} \, . \, P(x) \Rightarrow P(S(x)))] \\
\qquad \Rightarrow \\
\quad \forall x : \mathbf{N} \, . \, P(x)
\end{array}
$$

Recursive Function Definitions

- Recursion is extremely useful for defining functions
  - Can facilitate both reasoning and computation
- A faulty recursive definition may lead to inconsistencies
  - Example: $\forall n : \mathbf{N} \,.\, f(n) = f(n) + 1$
- There are several schemes for defining functions by recursion

Recursive Definition Schemes

- Each scheme has a set of **instance requirements**
- A scheme is **proper** if every instance of the scheme actually defines a function
- The **domain** of a scheme is the set of functions f such that f is definable by some instance of the scheme
- Designers of **mechanized mathematics systems** prefer schemes which:
  - Are proper
  - Have easily checked instance requirements
  - Have a large domain of useful functions

The Primitive Recursive Functions (1)

- The class $\mathcal{P}$ of **primitive recursive functions** is the smallest set of $f : \mathbf{N} \times \cdots \times \mathbf{N} \to \mathbf{N}$ closed under the following rules:

1. **Successor Function** $(\lambda\, x : \mathbf{N} \,.\, x + 1) \in \mathcal{P}$

2. **Constant Functions** Each $(\lambda\, x_1, \ldots, x_n : \mathbf{N} \,.\, m) \in \mathcal{P}$ where $0 \leq m, n$

3. **Projection Functions** Each $(\lambda\, x_1, \ldots, x_n : \mathbf{N} \,.\, x_i) \in \mathcal{P}$ where $1 \leq n$ and $1 \leq i \leq n$

4. **Composition** If $g_1, \ldots, g_m, h \in \mathcal{P}$, then $f \in \mathcal{P}$ where

$$\forall\, x_1, \ldots, x_n : \mathbf{N} \,.\, f(x_1, \ldots, x_n) = h(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n))$$

5. **Primitive Recursion** If $g, h \in \mathcal{P}$, then $f \in \mathcal{P}$ where

$$\forall\, x_2, \ldots, x_n : \mathbf{N} \,.\, f(0, x_2, \ldots, x_n) = g(x_2, \ldots, x_n)$$
$$\forall\, x_1, \ldots, x_n : \mathbf{N} \,.\, f(x_1 + 1, x_2, \ldots, x_n) = h(x_1, f(x_1, x_2, \ldots, x_n), x_2, \ldots, x_n)$$

The Primitive Recursive Functions (2)

- Example: The factorial function f : **N** $\rightarrow$ **N** is defined by:
  1. $f(0) = g() = 1$.
  2. $f(n+1) = h(n, f(n))$ where $h(x, y) = y * (x+1)$

- The primitive recursion scheme is proper

- $\mathcal{P}$ is a very large, but proper, subset of the computable total functions on **N**
  - $\mathcal{P}$ contains almost all functions on **N** commonly found in mathematics

- **Theorem** There exists a computable total function f : **N** $\rightarrow$ **N** such that $f \notin \mathcal{P}$

  Proof: Construct f by diagonalization

Well-Founded Relations

- A relation $R \subseteq A \times A$ is **well-founded**, if for all nonempty $B \subseteq A$, there is some $a \in B$ such that, for all $b \in B$, $\neg bRa$
  - a is called the R-least element of B

- **Theorem**. If $R \subseteq A \times A$ is a strict linear order, then R is well-founded iff R is a well-order.

Well-Founded Recursion

- A tuple (T, f, D, R) where
  - T is a theory
  - $f : A \to A$
  - D is a definition of the form

$$\forall\, x\ .\ f(x) = E(f(a_1(x)), \ldots, f(a_k(x)))$$

  - $R \subseteq A \times A$ is a well-founded relation

  defines f to be a total function in T by **well-founded recursion** if $a_i(x)\ R\ x$ for each i with $1 \le i \le k$

- Example: $(P, f, D, <)$ where
  - P is first-order Peano arithmetic
  - f : **N** $\to$ **N**
  - D is $\forall\, n\ .\ f(n) = \text{if}(n = 0, 1, f(n-1) * n)$
  - $<$ is the usual order on **N**

  defines the factorial function in P

Monotone Functionals

- A **functional** is an expression of type $\alpha \rightharpoonup \alpha$ where $\alpha = \alpha_1 \times \cdots \times \alpha_n \rightharpoonup \alpha_{n+1}$

- **Subfunction**: $\forall\, g, h : \alpha \,.\, g \sqsubseteq_\alpha h \equiv$
$$\forall\, x_1 : \alpha_1, \ldots, x_n : \alpha_n \,.\, g(x_1, \ldots, x_n)\!\downarrow \Rightarrow g(x_1, \ldots, x_n) = h(x_1, \ldots, x_n)$$

- **Monotone**: $\forall\, F : \alpha \rightharpoonup \alpha \,.\, \mathrm{monotone}_\alpha(F) \equiv$
$$\forall\, g, h : \alpha \,.\, g \sqsubseteq_\alpha h \Rightarrow F(g) \sqsubseteq_\alpha F(h)$$

- **Fixed Point Theorem**: Every monotone functional has a least fixed point.

  Proof: $F^\gamma(\triangle_\alpha)$ must be a fixed point for some ordinal γ, where $\triangle_\alpha$ is the empty function of type α

Monotone Functional Recursion

- A **recursive definition via a monotone functional** is a triple $R = (T, f, F)$ where:
  - $T = (L, \Gamma)$ is a theory (in a higher-order logic that admits partial functions)
  - f is a constant of type α which is not a member of L
  - F is a functional of type α which is monotone in T

- The **defining axiom** of R is φ which says "f is a least fixed point of F"

- The **definitional extension resulting from** R is the theory $(L \cup \{f\}, \Gamma \cup \{\varphi\})$

Examples

- **Factorial**: $\lambda f : \mathbf{N} \rightharpoonup \mathbf{N} \,.\, \lambda n : \mathbf{N} \,.\, \mathrm{if}(n = 0, 1, f(n-1) * n)$

- **Sum**: $\lambda \sigma : \mathbf{Z} \times \mathbf{Z} \times (\mathbf{Z} \rightharpoonup \mathbf{R}) \rightharpoonup \mathbf{R} \,.$
  $\lambda m, n : \mathbf{Z}, f : \mathbf{Z} \rightharpoonup \mathbf{R} \,.\, \mathrm{if}(m \leq n, \sigma(m, n-1, f) + f(n), 0)$

- **Empty function**: $\lambda f : \mathbf{Z} \rightharpoonup \mathbf{Z} \,.\, \lambda n : \mathbf{Z} \,.\, f(n)$

- **Empty function**: $\lambda f : \mathbf{Z} \rightharpoonup \mathbf{Z} \,.\, \lambda n : \mathbf{Z} \,.\, f(n) + 1$