Recursive Data Definition

example: *nat*

can be constructed by starting with 0 and repeatedly adding 1

construction axiom $\quad 0: nat$

construction axiom $\quad nat+1: nat$

	$\top$	by the axiom, $0: nat$
$\Rightarrow$	$0: nat$	add 1 to each side
$\Rightarrow$	$0+1: nat+1$	by arithmetic, $0+1 = 1$; by the axiom, $nat+1: nat$
$\Rightarrow$	$1: nat$	add 1 to each side
$\Rightarrow$	$1+1: nat+1$	by arithmetic, $1+1 = 2$; by the axiom, $nat+1: nat$
$\Rightarrow$	$2: nat$	and so on

Recursive Data Definition

example: *nat*

can be constructed by starting with 0 and repeatedly adding 1

construction axiom 0: *nat*

construction axiom *nat*+1: *nat*

nat = 0, 1, 2, 3, 4, 5, ... ?

nat = ..., -3, -2, -1, 0, 1, 2, 3, ... ?

nat = the rationals ?

nat = the reals ?

nat = 0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, ... ?

Recursive Data Definition

example: *nat*

can be constructed by starting with 0 and repeatedly adding 1

construction axiom $0{:}\ nat$

construction axiom $nat{+}1{:}\ nat$

induction axiom $0{:}\ B \;\wedge\; B{+}1{:}\ B \;\Rightarrow\; nat{:}\ B$

construction axiom $0,\ nat{+}1{:}\ nat$

induction axiom $0,\ B{+}1{:}\ B \;\Rightarrow\; nat{:}\ B$

construction axiom $P0 \wedge \forall n{:}\ nat\cdot\ Pn \Rightarrow P(n{+}1) \;\;\Leftarrow\;\; \forall n{:}\ nat\cdot\ Pn$

induction axiom $P0 \wedge \forall n{:}\ nat\cdot\ Pn \Rightarrow P(n{+}1) \;\;\Rightarrow\;\; \forall n{:}\ nat\cdot\ Pn$

Recursive Data Definition

nat induction

$$P0 \wedge \forall n{:}\ nat\cdot\ Pn \Rightarrow P(n{+}1) \quad\Longrightarrow\quad \forall n{:}\ nat\cdot\ Pn$$

$$P0 \vee \exists n{:}\ nat\cdot\ \neg Pn \wedge P(n{+}1) \quad\Longleftarrow\quad \exists n{:}\ nat\cdot\ Pn$$

$$\forall n{:}\ nat\cdot\ Pn \Rightarrow P(n{+}1) \quad\Longrightarrow\quad \forall n{:}\ nat\cdot\ (P0 \Rightarrow Pn)$$

$$\exists n{:}\ nat\cdot\ \neg Pn \wedge P(n{+}1) \quad\Longleftarrow\quad \exists n{:}\ nat\cdot\ (\neg P0 \wedge Pn)$$

$$\forall n{:}\ nat\cdot\ (\forall m{:}\ nat\cdot\ m<n \Rightarrow Pm) \Rightarrow Pn \quad\Longrightarrow\quad \forall n{:}\ nat\cdot\ Pn$$

$$\exists n{:}\ nat\cdot\ (\forall m{:}\ nat\cdot\ m<n \Rightarrow \neg Pm) \wedge Pn \quad\Longleftarrow\quad \exists n{:}\ nat\cdot\ Pn$$

philosophical induction: guessing the general case from special cases

(an important skill in mathematics)

philosophical deduction: proving, using the rules of logic

mathematical induction: an axiom (sometimes presented as a proof rule)

(mathematical induction is part of philosophical deduction)

Recursive Data Definition

example: *int*

Define $\quad int \;=\; nat, -nat$

or $\quad 0, int+1, int-1: int$

$\quad 0, B+1, B-1: B \;\Rightarrow\; int: B$

or $\quad P0 \land (\forall i: int\cdot Pi \Rightarrow P(i+1)) \land (\forall i: int\cdot Pi \Rightarrow P(i-1)) \;=\; \forall i: int\cdot Pi$

Recursive Data Definition

example: *pow*

Define $\qquad pow \;=\; 2^{nat}$

or $\qquad pow \;=\; \S p: nat\cdot \exists m: nat\cdot p = 2^m$

or $\qquad 1, 2{\times}pow: pow$

$\qquad 1, 2{\times}B: B \;\Rightarrow\; pow: B$

or $\qquad P1 \;\land\; \forall p: pow\cdot Pp \Rightarrow P(2{\times}p) \;=\; \forall p: pow\cdot Pp$

Least Fixed-Points

nat construction:	$0, nat{+}1 : nat$
nat induction:	$0, B{+}1 : B \;\Rightarrow\; nat : B$
nat fixed-point construction:	$nat \;=\; 0, nat{+}1$
nat fixed-point induction:	$B = 0, B{+}1 \;\Rightarrow\; nat : B$
x is a fixed-point of f	$x = f\,x$
grammar:	$exp \;=\;$ “x”, exp; “+”; exp
	$B \;=\;$ “x”, B; “+”; $B \;\Rightarrow\; exp : B$

Recursive Data Construction

$name$ = (expression involving $name$)

0. Construct — $name_0 = null$

   $name_{n+1}$ = (expression involving $name_n$)

1. Guess — $name_n$ = (expression involving n but not $name$)

2. Substitute ∞ for n — $name_\infty$ = (expression involving neither n nor $name$)

3. Test fixed-point — $name_\infty$ = (expression involving $name_\infty$)

4. Test least fixed-point — B = (expression involving B) $\Rightarrow$ $name_\infty : B$

Recursive Data Construction

example: *pow*

$$pow \;=\; 1, 2{\times}pow$$

0. Construct

$$pow_0 \;=\; null$$

$$pow_1 \;=\; 1, 2{\times}pow_0 \;=\; 1, 2{\times}null \;=\; 1, null \;=\; 1$$

$$pow_2 \;=\; 1, 2{\times}pow_1 \;=\; 1, 2{\times}1 \;=\; 1, 2$$

$$pow_3 \;=\; 1, 2{\times}pow_2 \;=\; 1, 2{\times}(1, 2) \;=\; 1, 2, 4$$

1. Guess $\qquad pow_n \;=\; 2^{0,..n}$

2. Substitute ∞ for n $\qquad pow_\infty \;=\; 2^{0,..\infty} \;=\; 2^{nat}$

Recursive Data Construction

example: pow

$$pow \;=\; 1, 2{\times}pow$$

3. Test fixed-point.

$$
\begin{array}{ll}
 & 2^{nat} = 1, 2{\times}2^{nat} \\
= & 2^{nat} = 2^0, 2^1{\times}2^{nat} \\
= & 2^{nat} = 2^0, 2^{1+nat} \\
= & 2^{nat} = 2^{0,\ 1+nat} \\
\Leftarrow & nat = 0, nat{+}1 \\
= & \top
\end{array}
$$

Recursive Data Construction

example: *pow*

$$pow \;=\; 1, 2{\times}pow$$

4. Test least fixed-point

$2^{nat}: B$

$=$ $\forall n: nat\cdot\ 2^n: B$ — use nat induction with $Pn = 2^n: B$

$\Leftarrow$ $2^0: B \;\wedge\; \forall n: nat\cdot\ 2^n: B \;\Rightarrow\; 2^{n+1}: B$ — change variable

$=$ $1: B \;\wedge\; \forall m: 2^{nat}\cdot\ m: B \;\Rightarrow\; 2{\times}m: B$ — increase domain

$\Leftarrow$ $1: B \;\wedge\; \forall m: nat\cdot\ m: B \;\Rightarrow\; 2{\times}m: B$ — domain change law

$=$ $1: B \;\wedge\; \forall m: nat`B\cdot\ 2{\times}m: B$ — increase domain

$\Leftarrow$ $1: B \;\wedge\; \forall m: B\cdot\ 2{\times}m: B$

$\Leftarrow$ $B = 1, 2{\times}B$

Recursive Data Construction

Alternative step 0: instead of *null* use

$name_0 = whatever$

Alternative step 2: instead of $name_\infty$ use

§$x\cdot$ $LIM\ n\cdot x: name_n$

Recursive Specification Definition

$$zap \;=\; \textbf{if } x{=}0 \textbf{ then } y{:=}\,0 \textbf{ else } x{:=}\,x{-}1.\; t{:=}\,t{+}1.\; zap \textbf{ fi}$$

solutions

(a) $x{\ge}0 \;\Rightarrow\; x'{=}y'{=}0 \wedge t' = t{+}x$

(b) $\textbf{if } x{\ge}0 \textbf{ then } x'{=}y'{=}0 \wedge t' = t{+}x \textbf{ else } t'{=}\infty \textbf{ fi}$

(c) $x'{=}y'{=}0 \wedge (x{\ge}0 \Rightarrow t' = t{+}x)$

(d) $x'{=}y'{=}0 \wedge \textbf{if } x{\ge}0 \textbf{ then } t' = t{+}x \textbf{ else } t'{=}\infty \textbf{ fi}$

(e) $x'{=}y'{=}0 \wedge t' = t{+}x$

(f) $x{\ge}0 \wedge x'{=}y'{=}0 \wedge t' = t{+}x$

$$\begin{tikzcd}
 & \text{(a)} & & \\
\text{(b)} \arrow[ur, Rightarrow] & & \text{(c)} \arrow[ul, Rightarrow] & \\
 & \text{(d)} \arrow[ul, Rightarrow] \arrow[ur, Rightarrow] & & \text{(e)} \arrow[ul, Rightarrow] \\
 & & \text{(f)} \arrow[ul, Rightarrow] \arrow[ur, Rightarrow] &
\end{tikzcd}$$

$$x{\ge}0 \;\Rightarrow\; x'{=}y'{=}0 \wedge t' = t{+}x \;\Leftarrow\; zap$$

$$zap \;\Leftarrow\; \textbf{if } x{=}0 \textbf{ then } y{:=}\,0 \textbf{ else } x{:=}\,x{-}1.\; t{:=}\,t{+}1.\; zap \textbf{ fi}$$

Recursive Specification Definition

zap construction

$t' \geq t \ \Leftarrow\ zap$

if $x=0$ **then** $y:= 0$ **else** $x:= x-1.\ \ t:= t+1.\ \ zap$ **fi** $\ \Leftarrow\ zap$

nat construction

$0: nat$

$nat+1: nat$

Recursive Specification Definition

zap construction

$$t' \geq t \wedge \textbf{if } x=0 \textbf{ then } y:=0 \textbf{ else } x:=x-1.\ t:=t+1.\ zap \textbf{ fi} \Longleftarrow zap$$

zap induction

$$\forall \sigma, \sigma' \cdot t' \geq t \wedge \textbf{if } x=0 \textbf{ then } y:=0 \textbf{ else } x:=x-1.\ t:=t+1.\ P \textbf{ fi} \Leftarrow P$$

$$\Longrightarrow \quad \forall \sigma, \sigma' \cdot zap \Leftarrow P$$

nat construction

$$0, nat+1 : nat$$

nat induction

$$0, B+1 : B \quad \Longrightarrow \quad nat : B$$

Recursive Specification Definition

zap construction

$$t' \geq t \wedge \textbf{if } x{=}0 \textbf{ then } y{:=}\ 0 \textbf{ else } x{:=}\ x{-}1.\ t{:=}\ t{+}1.\ zap \textbf{ fi} \ \Leftarrow\ zap$$

zap induction

$$\forall \sigma, \sigma' \cdot\ t' \geq t \wedge \textbf{if } x{=}0 \textbf{ then } y{:=}\ 0 \textbf{ else } x{:=}\ x{-}1.\ t{:=}\ t{+}1.\ P \textbf{ fi} \ \Leftarrow\ P$$

$$\Longrightarrow \quad \forall \sigma, \sigma' \cdot\ zap \Leftarrow P$$

zap fixed-point construction

$$zap \ = \ t' \geq t \wedge \textbf{if } x{=}0 \textbf{ then } y{:=}\ 0 \textbf{ else } x{:=}\ x{-}1.\ t{:=}\ t{+}1.\ zap \textbf{ fi}$$

zap fixed-point induction

$$\forall \sigma, \sigma' \cdot\ (P \ = \ t' \geq t \wedge \textbf{if } x{=}0 \textbf{ then } y{:=}\ 0 \textbf{ else } x{:=}\ x{-}1.\ t{:=}\ t{+}1.\ P \textbf{ fi})$$

$$\Longrightarrow \quad \forall \sigma, \sigma' \cdot\ zap \Leftarrow P$$

Recursive Specification Construction

$$zap \;=\; \textbf{if } x{=}0 \textbf{ then } y{:=}\,0 \textbf{ else } x{:=}\,x{-}1.\; t{:=}\,t{+}1.\; zap \textbf{ fi}$$

$$zap_0 \;=\; \top$$

$$zap_1 \;=\; \textbf{if } x{=}0 \textbf{ then } y{:=}\,0 \textbf{ else } x{:=}\,x{-}1.\; t{:=}\,t{+}1.\; zap_0 \textbf{ fi}$$

$$=\; x{=}0 \;\Rightarrow\; x'{=}y'{=}0 \land t'{=}t$$

$$zap_2 \;=\; \textbf{if } x{=}0 \textbf{ then } y{:=}\,0 \textbf{ else } x{:=}\,x{-}1.\; t{:=}\,t{+}1.\; zap_1 \textbf{ fi}$$

$$=\; 0{\le}x{<}2 \;\Rightarrow\; x'{=}y'{=}0 \land t' = t{+}x$$

$$zap_n \;=\; 0{\le}x{<}n \;\Rightarrow\; x'{=}y'{=}0 \land t' = t{+}x$$

$$zap_\infty \;=\; 0{\le}x{<}\infty \;\Rightarrow\; x'{=}y'{=}0 \land t' = t{+}x$$

Recursive Specification Construction

Alternative step 0: instead of $\top$ use

$$name_0 = whatever$$

Alternative step 2: instead of $name_\infty$ use

$$LIM\ n \cdot name_n$$

Recursive Specification Construction

$$zap \quad = \quad \textbf{if } x{=}0 \textbf{ then } y{:=}\ 0 \textbf{ else } x{:=}\ x{-}1.\ t{:=}\ t{+}1.\ zap \textbf{ fi}$$

$$zap_0 \quad = \quad t' \ge t$$

$$zap_1 \quad = \quad \textbf{if } x{=}0 \textbf{ then } y{:=}\ 0 \textbf{ else } x{:=}\ x{-}1.\ t{:=}\ t{+}1.\ zap_0 \textbf{ fi}$$

$$\quad = \quad \textbf{if } x{=}0 \textbf{ then } x'{=}y'{=}0 \wedge t'{=}t \textbf{ else } t' \ge t{+}1 \textbf{ fi}$$

$$zap_2 \quad = \quad \textbf{if } x{=}0 \textbf{ then } y{:=}\ 0 \textbf{ else } x{:=}\ x{-}1.\ t{:=}\ t{+}1.\ zap_1 \textbf{ fi}$$

$$\quad = \quad \textbf{if } 0{\le}x{<}2 \textbf{ then } x'{=}y'{=}0 \wedge t' = t{+}x \textbf{ else } t' \ge t{+}2 \textbf{ fi}$$

$$zap_n \quad = \quad \textbf{if } 0{\le}x{<}n \textbf{ then } x'{=}y'{=}0 \wedge t'{=}t{+}x \textbf{ else } t' \ge t{+}n \textbf{ fi}$$

$$zap_\infty \quad = \quad \textbf{if } 0{\le}x \textbf{ then } x'{=}y'{=}0 \wedge t'{=}t{+}x \textbf{ else } t'{=}\infty \textbf{ fi}$$

Loop Definition

while-loop construction

$$t' \ge t \wedge \mathbf{if}\ b\ \mathbf{then}\ P.\ \ t{:=}\ t{+}1.\ \ \mathbf{while}\ b\ \mathbf{do}\ P\ \mathbf{od}\ \mathbf{else}\ ok\ \mathbf{fi} \quad \Leftarrow \quad \mathbf{while}\ b\ \mathbf{do}\ P\ \mathbf{od}$$

while-loop induction

$$\forall \sigma, \sigma' \cdot\ t' \ge t \wedge \mathbf{if}\ b\ \mathbf{then}\ P.\ \ t{:=}\ t{+}1.\ \ W\ \mathbf{else}\ ok\ \mathbf{fi} \Leftarrow W$$

$$\Longrightarrow \quad \forall \sigma, \sigma' \cdot\ \mathbf{while}\ b\ \mathbf{do}\ P\ \mathbf{od} \Leftarrow W$$

while-loop fixed-point construction

$$\mathbf{while}\ b\ \mathbf{do}\ P\ \mathbf{od} \quad = \quad t' \ge t \wedge \mathbf{if}\ b\ \mathbf{then}\ P.\ \ t{:=}\ t{+}1.\ \ \mathbf{while}\ b\ \mathbf{do}\ P\ \mathbf{od}\ \mathbf{else}\ ok\ \mathbf{fi}$$

while-loop fixed-point induction

$$\forall \sigma, \sigma' \cdot\ (P \ = \ t' \ge t \wedge \mathbf{if}\ b\ \mathbf{then}\ P.\ \ t{:=}\ t{+}1.\ \ W\ \mathbf{else}\ ok\ \mathbf{fi})$$

$$\Longrightarrow \quad \forall \sigma, \sigma' \cdot\ \mathbf{while}\ b\ \mathbf{do}\ P\ \mathbf{od} \Leftarrow W$$