JOURNAL OF FORMALIZED MATHEMATICS
Volume 1, Released 1989, Published 2003
Inst. of Computer Science, Univ. of Białystok

Recursive Definitions

Krzysztof Hryniewiecki
Warsaw University

Summary. The text contains some schemes which allow elimination of definitions by recursion.

MML Identifier: RECDEF_1.

WWW: http://mizar.org/JFM/Vol1/recdef_1.html

The articles [6], [5], [8], [7], [1], [9], [3], [2], and [4] provide the notation and terminology for this paper.

We use the following convention: n, k denote natural numbers, x, y, z, y_1, y_2 denote sets, and p denotes a finite sequence.

Let D be a set, let p be a partial function from D to $\mathbb{N}$, and let n be an element of D. Then $p(n)$ is a natural number.

In this article we present several logical schemes. The scheme *RecEx* deals with a set $\mathcal{A}$ and a ternary predicate $\mathcal{P}$, and states that:

There exists a function f such that $\mathrm{dom}\, f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every element n of $\mathbb{N}$ holds $\mathcal{P}[n, f(n), f(n+1)]$

provided the parameters meet the following requirements:

- For every natural number n and for every set x there exists a set y such that $\mathcal{P}[n,x,y]$, and
- For every natural number n and for all sets x, y_1, y_2 such that $\mathcal{P}[n,x,y_1]$ and $\mathcal{P}[n,x,y_2]$ holds $y_1 = y_2$.

The scheme *RecExD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, and a ternary predicate $\mathcal{P}$, and states that:

There exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $f(0) = \mathcal{B}$ and for every element n of $\mathbb{N}$ holds $\mathcal{P}[n, f(n), f(n+1)]$

provided the parameters have the following property:

- For every natural number n and for every element x of $\mathcal{A}$ there exists an element y of $\mathcal{A}$ such that $\mathcal{P}[n,x,y]$.

The scheme *LambdaRecEx* deals with a set $\mathcal{A}$ and a binary functor $\mathcal{F}$ yielding a set, and states that:

There exists a function f such that $\mathrm{dom}\, f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every element n of $\mathbb{N}$ holds $f(n+1) = \mathcal{F}(n, f(n))$

for all values of the parameters.

The scheme *LambdaRecExD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, and a binary functor $\mathcal{F}$ yielding an element of $\mathcal{A}$, and states that:

There exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $f(0) = \mathcal{B}$ and for every element n of $\mathbb{N}$ holds $f(n+1) = \mathcal{F}(n, f(n))$

for all values of the parameters.

The scheme *FinRecEx* deals with a set $\mathcal{A}$, a natural number $\mathcal{B}$, and a ternary predicate $\mathcal{P}$, and states that:

There exists a finite sequence p such that $\operatorname{len} p = \mathcal{B}$ but $p(1) = \mathcal{A}$ or $\mathcal{B} = 0$ but for every n such that $1 \leq n$ and $n < \mathcal{B}$ holds $\mathcal{P}[n, p(n), p(n+1)]$

provided the parameters meet the following conditions:

- For every natural number n such that $1 \leq n$ and $n < \mathcal{B}$ and for every set x there exists a set y such that $\mathcal{P}[n,x,y]$, and
- For every natural number n such that $1 \leq n$ and $n < \mathcal{B}$ and for all sets x, y_1, y_2 such that $\mathcal{P}[n,x,y_1]$ and $\mathcal{P}[n,x,y_2]$ holds $y_1 = y_2$.

The scheme *FinRecExD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, a natural number $\mathcal{C}$, and a ternary predicate $\mathcal{P}$, and states that:

There exists a finite sequence p of elements of $\mathcal{A}$ such that $\operatorname{len} p = \mathcal{C}$ but $p(1) = \mathcal{B}$ or $\mathcal{C} = 0$ but for every n such that $1 \leq n$ and $n < \mathcal{C}$ holds $\mathcal{P}[n, p(n), p(n+1)]$

provided the parameters have the following property:

- Let n be a natural number. Suppose $1 \leq n$ and $n < \mathcal{C}$. Let x be an element of $\mathcal{A}$. Then there exists an element y of $\mathcal{A}$ such that $\mathcal{P}[n,x,y]$.

The scheme *SeqBinOpEx* deals with a finite sequence $\mathcal{A}$ and a ternary predicate $\mathcal{P}$, and states that:

There exists x and there exists a finite sequence p such that $x = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $\mathcal{P}[\mathcal{A}(k+1), p(k), p(k+1)]$

provided the following conditions are met:

- For all k, x such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ there exists y such that $\mathcal{P}[\mathcal{A}(k+1),x,y]$, and
- For all k, x, y_1, y_2, z such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ and $z = \mathcal{A}(k+1)$ and $\mathcal{P}[z,x,y_1]$ and $\mathcal{P}[z,x,y_2]$ holds $y_1 = y_2$.

The scheme *LambdaSeqBinOpEx* deals with a finite sequence $\mathcal{A}$ and a binary functor $\mathcal{F}$ yielding a set, and states that:

There exists x and there exists a finite sequence p such that $x = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $p(k+1) = \mathcal{F}(\mathcal{A}(k+1), p(k))$

for all values of the parameters.

The scheme *RecUn* deals with a set $\mathcal{A}$, functions $\mathcal{B}$, $\mathcal{C}$, and a ternary predicate $\mathcal{P}$, and states that:

$\mathcal{B} = \mathcal{C}$

provided the parameters satisfy the following conditions:

- $\operatorname{dom} \mathcal{B} = \mathbb{N}$ and $\mathcal{B}(0) = \mathcal{A}$ and for every n holds $\mathcal{P}[n, \mathcal{B}(n), \mathcal{B}(n+1)]$,
- $\operatorname{dom} \mathcal{C} = \mathbb{N}$ and $\mathcal{C}(0) = \mathcal{A}$ and for every n holds $\mathcal{P}[n, \mathcal{C}(n), \mathcal{C}(n+1)]$, and
- For every n and for all sets x, y_1, y_2 such that $\mathcal{P}[n,x,y_1]$ and $\mathcal{P}[n,x,y_2]$ holds $y_1 = y_2$.

The scheme *RecUnD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, functions $\mathcal{C}$, $\mathcal{D}$ from $\mathbb{N}$ into $\mathcal{A}$, and a ternary predicate $\mathcal{P}$, and states that:

$\mathcal{C} = \mathcal{D}$

provided the parameters meet the following requirements:

- $\mathcal{C}(0) = \mathcal{B}$ and for every n holds $\mathcal{P}[n, \mathcal{C}(n), \mathcal{C}(n+1)]$,
- $\mathcal{D}(0) = \mathcal{B}$ and for every n holds $\mathcal{P}[n, \mathcal{D}(n), \mathcal{D}(n+1)]$, and
- For every natural number n and for all elements x, y_1, y_2 of $\mathcal{A}$ such that $\mathcal{P}[n,x,y_1]$ and $\mathcal{P}[n,x,y_2]$ holds $y_1 = y_2$.

The scheme *LambdaRecUn* deals with a set $\mathcal{A}$, a binary functor $\mathcal{F}$ yielding a set, and functions $\mathcal{B}$, $\mathcal{C}$, and states that:

$\mathcal{B} = \mathcal{C}$

provided the parameters meet the following conditions:

- $\operatorname{dom} \mathcal{B} = \mathbb{N}$ and $\mathcal{B}(0) = \mathcal{A}$ and for every n holds $\mathcal{B}(n+1) = \mathcal{F}(n, \mathcal{B}(n))$, and
- $\operatorname{dom} \mathcal{C} = \mathbb{N}$ and $\mathcal{C}(0) = \mathcal{A}$ and for every n holds $\mathcal{C}(n+1) = \mathcal{F}(n, \mathcal{C}(n))$.

The scheme *LambdaRecUnD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, a binary functor $\mathcal{F}$ yielding an element of $\mathcal{A}$, and functions $\mathcal{C}$, $\mathcal{D}$ from $\mathbb{N}$ into $\mathcal{A}$, and states that:

$\mathcal{C} = \mathcal{D}$

provided the following conditions are satisfied:

- $\mathcal{C}(0) = \mathcal{B}$ and for every n holds $\mathcal{C}(n+1) = \mathcal{F}(n, \mathcal{C}(n))$, and

- $\mathcal{D}(0) = \mathcal{B}$ and for every n holds $\mathcal{D}(n+1) = \mathcal{F}(n, \mathcal{D}(n))$.

The scheme *LambdaRecUnR* deals with a real number $\mathcal{A}$, a binary functor $\mathcal{F}$ yielding a set, and functions $\mathcal{B}$, $\mathcal{C}$ from $\mathbb{N}$ into $\mathbb{R}$, and states that:

$\mathcal{B} = \mathcal{C}$

provided the parameters meet the following conditions:

- $\mathcal{B}(0) = \mathcal{A}$ and for every n holds $\mathcal{B}(n+1) = \mathcal{F}(n, \mathcal{B}(n))$, and
- $\mathcal{C}(0) = \mathcal{A}$ and for every n holds $\mathcal{C}(n+1) = \mathcal{F}(n, \mathcal{C}(n))$.

The scheme *FinRecUn* deals with a set $\mathcal{A}$, a natural number $\mathcal{B}$, finite sequences $\mathcal{C}$, $\mathcal{D}$, and a ternary predicate $\mathcal{P}$, and states that:

$\mathcal{C} = \mathcal{D}$

provided the parameters meet the following requirements:

- For every n such that $1 \leq n$ and $n < \mathcal{B}$ and for all sets x, y_1, y_2 such that $\mathcal{P}[n, x, y_1]$ and $\mathcal{P}[n, x, y_2]$ holds $y_1 = y_2$,
- $\operatorname{len} \mathcal{C} = \mathcal{B}$ but $\mathcal{C}(1) = \mathcal{A}$ or $\mathcal{B} = 0$ but for every n such that $1 \leq n$ and $n < \mathcal{B}$ holds $\mathcal{P}[n, \mathcal{C}(n), \mathcal{C}(n+1)]$, and
- $\operatorname{len} \mathcal{D} = \mathcal{B}$ but $\mathcal{D}(1) = \mathcal{A}$ or $\mathcal{B} = 0$ but for every n such that $1 \leq n$ and $n < \mathcal{B}$ holds $\mathcal{P}[n, \mathcal{D}(n), \mathcal{D}(n+1)]$.

The scheme *FinRecUnD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, a natural number $\mathcal{C}$, finite sequences $\mathcal{D}$, $\mathcal{E}$ of elements of $\mathcal{A}$, and a ternary predicate $\mathcal{P}$, and states that:

$\mathcal{D} = \mathcal{E}$

provided the parameters meet the following requirements:

- For every n such that $1 \leq n$ and $n < \mathcal{C}$ and for all elements x, y_1, y_2 of $\mathcal{A}$ such that $\mathcal{P}[n, x, y_1]$ and $\mathcal{P}[n, x, y_2]$ holds $y_1 = y_2$,
- $\operatorname{len} \mathcal{D} = \mathcal{C}$ but $\mathcal{D}(1) = \mathcal{B}$ or $\mathcal{C} = 0$ but for every n such that $1 \leq n$ and $n < \mathcal{C}$ holds $\mathcal{P}[n, \mathcal{D}(n), \mathcal{D}(n+1)]$, and
- $\operatorname{len} \mathcal{E} = \mathcal{C}$ but $\mathcal{E}(1) = \mathcal{B}$ or $\mathcal{C} = 0$ but for every n such that $1 \leq n$ and $n < \mathcal{C}$ holds $\mathcal{P}[n, \mathcal{E}(n), \mathcal{E}(n+1)]$.

The scheme *SeqBinOpUn* deals with a finite sequence $\mathcal{A}$, sets $\mathcal{B}$, $\mathcal{C}$, and a ternary predicate $\mathcal{P}$, and states that:

$\mathcal{B} = \mathcal{C}$

provided the following conditions are satisfied:

- For all k, x, y_1, y_2, z such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ and $z = \mathcal{A}(k+1)$ and $\mathcal{P}[z, x, y_1]$ and $\mathcal{P}[z, x, y_2]$ holds $y_1 = y_2$,
- There exists a finite sequence p such that $\mathcal{B} = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $\mathcal{P}[\mathcal{A}(k+1), p(k), p(k+1)]$, and
- There exists a finite sequence p such that $\mathcal{C} = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $\mathcal{P}[\mathcal{A}(k+1), p(k), p(k+1)]$.

The scheme *LambdaSeqBinOpUn* deals with a finite sequence $\mathcal{A}$, a binary functor $\mathcal{F}$ yielding a set, and sets $\mathcal{B}$, $\mathcal{C}$, and states that:

$\mathcal{B} = \mathcal{C}$

provided the parameters meet the following conditions:

- There exists a finite sequence p such that $\mathcal{B} = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $p(k+1) = \mathcal{F}(\mathcal{A}(k+1), p(k))$, and
- There exists a finite sequence p such that $\mathcal{C} = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $p(k+1) = \mathcal{F}(\mathcal{A}(k+1), p(k))$.

The scheme *DefRec* deals with a set $\mathcal{A}$, a natural number $\mathcal{B}$, and a ternary predicate $\mathcal{P}$, and states that:

(i) There exists a set y and there exists a function f such that $y = f(\mathcal{B})$ and $\operatorname{dom} f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every n holds $\mathcal{P}[n, f(n), f(n+1)]$, and

(ii) for all sets y_1, y_2 such that there exists a function f such that $y_1 = f(\mathcal{B})$ and $\operatorname{dom} f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every n holds $\mathcal{P}[n, f(n), f(n+1)]$ and there exists a function f such that $y_2 = f(\mathcal{B})$ and $\operatorname{dom} f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every n holds $\mathcal{P}[n, f(n), f(n+1)]$ holds $y_1 = y_2$

provided the following conditions are satisfied:

- For all n, x there exists y such that $\mathcal{P}[n,x,y]$, and
- For all n, x, y_1, y_2 such that $\mathcal{P}[n,x,y_1]$ and $\mathcal{P}[n,x,y_2]$ holds $y_1 = y_2$.

The scheme *LambdaDefRec* deals with a set $\mathcal{A}$, a natural number $\mathcal{B}$, and a binary functor $\mathcal{F}$ yielding a set, and states that:

(i) There exists a set y and there exists a function f such that $y = f(\mathcal{B})$ and $\operatorname{dom} f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every n holds $f(n+1) = \mathcal{F}(n, f(n))$, and

(ii) for all sets y_1, y_2 such that there exists a function f such that $y_1 = f(\mathcal{B})$ and $\operatorname{dom} f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every n holds $f(n+1) = \mathcal{F}(n, f(n))$ and there exists a function f such that $y_2 = f(\mathcal{B})$ and $\operatorname{dom} f = \mathbb{N}$ and $f(0) = \mathcal{A}$ and for every n holds $f(n+1) = \mathcal{F}(n, f(n))$ holds $y_1 = y_2$

for all values of the parameters.

The scheme *DefRecD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, a natural number $\mathcal{C}$, and a ternary predicate $\mathcal{P}$, and states that:

(i) There exists an element y of $\mathcal{A}$ and there exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $y = f(\mathcal{C})$ and $f(0) = \mathcal{B}$ and for every n holds $\mathcal{P}[n, f(n), f(n+1)]$, and

(ii) for all elements y_1, y_2 of $\mathcal{A}$ such that there exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $y_1 = f(\mathcal{C})$ and $f(0) = \mathcal{B}$ and for every n holds $\mathcal{P}[n, f(n), f(n+1)]$ and there exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $y_2 = f(\mathcal{C})$ and $f(0) = \mathcal{B}$ and for every n holds $\mathcal{P}[n, f(n), f(n+1)]$ holds $y_1 = y_2$

provided the following conditions are met:

- For every natural number n and for every element x of $\mathcal{A}$ there exists an element y of $\mathcal{A}$ such that $\mathcal{P}[n,x,y]$, and
- For every natural number n and for all elements x, y_1, y_2 of $\mathcal{A}$ such that $\mathcal{P}[n,x,y_1]$ and $\mathcal{P}[n,x,y_2]$ holds $y_1 = y_2$.

The scheme *LambdaDefRecD* deals with a non empty set $\mathcal{A}$, an element $\mathcal{B}$ of $\mathcal{A}$, a natural number $\mathcal{C}$, and a binary functor $\mathcal{F}$ yielding an element of $\mathcal{A}$, and states that:

(i) There exists an element y of $\mathcal{A}$ and there exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $y = f(\mathcal{C})$ and $f(0) = \mathcal{B}$ and for every natural number n holds $f(n+1) = \mathcal{F}(n, f(n))$, and

(ii) for all elements y_1, y_2 of $\mathcal{A}$ such that there exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $y_1 = f(\mathcal{C})$ and $f(0) = \mathcal{B}$ and for every natural number n holds $f(n+1) = \mathcal{F}(n, f(n))$ and there exists a function f from $\mathbb{N}$ into $\mathcal{A}$ such that $y_2 = f(\mathcal{C})$ and $f(0) = \mathcal{B}$ and for every natural number n holds $f(n+1) = \mathcal{F}(n, f(n))$ holds $y_1 = y_2$

for all values of the parameters.

The scheme *SeqBinOpDef* deals with a finite sequence $\mathcal{A}$ and a ternary predicate $\mathcal{P}$, and states that:

(i) There exists x and there exists a finite sequence p such that $x = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $\mathcal{P}[\mathcal{A}(k+1), p(k), p(k+1)]$, and

(ii) for all x, y such that there exists a finite sequence p such that $x = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $\mathcal{P}[\mathcal{A}(k+1), p(k), p(k+1)]$ and there exists a finite sequence p such that $y = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $\mathcal{P}[\mathcal{A}(k+1), p(k), p(k+1)]$ holds $x = y$

provided the following conditions are satisfied:

- For all k, y such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ there exists z such that $\mathcal{P}[\mathcal{A}(k+1), y, z]$, and
- For all k, x, y_1, y_2, z such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ and $z = \mathcal{A}(k+1)$ and $\mathcal{P}[z,x,y_1]$ and $\mathcal{P}[z,x,y_2]$ holds $y_1 = y_2$.

The scheme *LambdaSeqBinOpDef* deals with a finite sequence $\mathcal{A}$ and a binary functor $\mathcal{F}$ yielding a set, and states that:

(i) There exists x and there exists a finite sequence p such that $x = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $p(k+1) = \mathcal{F}(\mathcal{A}(k+1), p(k))$, and

(ii) for all x, y such that there exists a finite sequence p such that $x = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$

holds $p(k+1) = \mathcal{F}(\mathcal{A}(k+1), p(k))$ and there exists a finite sequence p such that $y = p(\operatorname{len} p)$ and $\operatorname{len} p = \operatorname{len} \mathcal{A}$ and $p(1) = \mathcal{A}(1)$ and for every k such that $1 \leq k$ and $k < \operatorname{len} \mathcal{A}$ holds $p(k+1) = \mathcal{F}(\mathcal{A}(k+1), p(k))$ holds $x = y$

for all values of the parameters.

REFERENCES

[1] Grzegorz Bancerek. The fundamental properties of natural numbers. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/nat_1.html`.

[2] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/finseq_1.html`.

[3] Czesław Byliński. Functions and their basic properties. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/funct_1.html`.

[4] Czesław Byliński. Functions from a set to a set. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/funct_2.html`.

[5] Czesław Byliński. Some basic properties of sets. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/zfmisc_1.html`.

[6] Andrzej Trybulec. Tarski Grothendieck set theory. *Journal of Formalized Mathematics*, Axiomatics, 1989. `http://mizar.org/JFM/Axiomatics/tarski.html`.

[7] Andrzej Trybulec. Subsets of real numbers. *Journal of Formalized Mathematics*, Addenda, 2003. `http://mizar.org/JFM/Addenda/numbers.html`.

[8] Zinaida Trybulec. Properties of subsets. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/subset_1.html`.

[9] Edmund Woronowicz. Relations and their basic properties. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/relat_1.html`.

Received September 4, 1989

Published January 2, 2004