FORMALIZED MATHEMATICS
Volume 6, Number 1, 1997
Warsaw University - Białystok

Memory Handling for $\mathbf{SCM}_{\mathrm{FSA}}$ [1]

Piotr Rudnicki
University of Alberta
Edmonton

Andrzej Trybulec
Warsaw University
Białystok

Summary. We introduce some terminology for reasoning about memory used in programs in general and in macro instructions (introduced in [26]) in particular. The usage of integer locations and of finite sequence locations by a program is treated separately. We define some functors for selecting memory locations needed for local (temporary) variables in macro instructions. Some semantic properties of the introduced notions are given in terms of executions of macro instructions.

MML Identifier: `SF_MASTR`.

The articles [21], [31], [19], [12], [30], [22], [14], [2], [28], [15], [20], [6], [13], [1], [3], [17], [11], [4], [7], [29], [32], [8], [9], [10], [5], [16], [25], [18], [27], [23], [24], and [26] provide the terminology and notation for this paper.

1. Preliminaries

One can prove the following three propositions:

(1) For all sets x, y, a and for every function f such that $f(x) = f(y)$ holds $f(a) = (f \cdot (\mathrm{id}_{\mathrm{dom}\, f} +\cdot (x, y)))(a)$.

(2) For all sets x, y and for every function f such that if $x \in \mathrm{dom}\, f$, then $y \in \mathrm{dom}\, f$ and $f(x) = f(y)$ holds $f = f \cdot (\mathrm{id}_{\mathrm{dom}\, f} +\cdot (x, y))$.

(3) For all sets A, B and for every function f from A into B holds $\mathrm{dom}\, f \subseteq A$.

Let A be a finite set and let B be a set. Note that every function from A into B is finite.

Let A be a finite set, let B be a set, and let f be a function from A into $\mathrm{Fin}\, B$. Observe that $\mathrm{Union}\, f$ is finite.

[1]This work was partially supported by NSERC Grant OGP9207 and NATO CRG 951368.

ISSN 1426–2630

In the sequel N will be a non empty set with non empty elements.

The following proposition is true

(4) Let S be a definite AMI over N and let p be a programmed finite partial state of S. Then $\operatorname{rng} p \subseteq$ the instructions of S.

Let us mention that Int-Locations is non empty.

Let us mention that FinSeq-Locations is non empty.

2. Uniqueness of instruction components

For simplicity we adopt the following rules: a, b, c, a_1, a_2, b_1, b_2 will be integer locations, l, l_1, l_2 will be instructions-locations of $\mathbf{SCM}_{\mathrm{FSA}}$, f, f_1, f_2 will be finite sequence locations, and i, j will be instructions of $\mathbf{SCM}_{\mathrm{FSA}}$.

The following propositions are true:

(5) If $a_1{:=}b_1 = a_2{:=}b_2$, then $a_1 = a_2$ and $b_1 = b_2$.

(6) If $\mathrm{AddTo}(a_1, b_1) = \mathrm{AddTo}(a_2, b_2)$, then $a_1 = a_2$ and $b_1 = b_2$.

(7) If $\mathrm{SubFrom}(a_1, b_1) = \mathrm{SubFrom}(a_2, b_2)$, then $a_1 = a_2$ and $b_1 = b_2$.

(8) If $\mathrm{MultBy}(a_1, b_1) = \mathrm{MultBy}(a_2, b_2)$, then $a_1 = a_2$ and $b_1 = b_2$.

(9) If $\mathrm{Divide}(a_1, b_1) = \mathrm{Divide}(a_2, b_2)$, then $a_1 = a_2$ and $b_1 = b_2$.

(10) If goto l_1 = goto l_2, then $l_1 = l_2$.

(11) If **if** $a_1 = 0$ **goto** l_1 = **if** $a_2 = 0$ **goto** l_2, then $a_1 = a_2$ and $l_1 = l_2$.

(12) If **if** $a_1 > 0$ **goto** l_1 = **if** $a_2 > 0$ **goto** l_2, then $a_1 = a_2$ and $l_1 = l_2$.

(13) If $b_1{:=}{f_1}_{a_1} = b_2{:=}{f_2}_{a_2}$, then $a_1 = a_2$ and $b_1 = b_2$ and $f_1 = f_2$.

(14) If ${f_1}_{a_1}{:=}b_1 = {f_2}_{a_2}{:=}b_2$, then $a_1 = a_2$ and $b_1 = b_2$ and $f_1 = f_2$.

(15) If $a_1{:=}\mathrm{len} f_1 = a_2{:=}\mathrm{len} f_2$, then $a_1 = a_2$ and $f_1 = f_2$.

(16) If $f_1{:=}\langle\underbrace{0, \ldots, 0}_{a_1}\rangle = f_2{:=}\langle\underbrace{0, \ldots, 0}_{a_2}\rangle$, then $a_1 = a_2$ and $f_1 = f_2$.

3. Integer locations used in macros

Let i be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$. The functor $\mathrm{UsedIntLoc}(i)$ yields an element of Fin Int-Locations and is defined as follows:

(Def. 1) (i) There exist integer locations a, b such that $i = a{:=}b$ or $i = \mathrm{AddTo}(a, b)$ or $i = \mathrm{SubFrom}(a, b)$ or $i = \mathrm{MultBy}(a, b)$ or $i = \mathrm{Divide}(a, b)$ but $\mathrm{UsedIntLoc}(i) = \{a, b\}$ if $\mathrm{InsCode}(i) \in \{1, 2, 3, 4, 5\}$,

(ii) there exists an integer location a and there exists an instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ such that $i =$ **if** $a = 0$ **goto** l or $i =$ **if** $a > 0$ **goto** l but $\mathrm{UsedIntLoc}(i) = \{a\}$ if $\mathrm{InsCode}(i) = 7$ or $\mathrm{InsCode}(i) = 8$,

(iii) there exist integer locations a, b and there exists a finite sequence location f such that $i = b{:=}f_a$ or $i = f_a{:=}b$ but $\mathrm{UsedIntLoc}(i) = \{a, b\}$ if $\mathrm{InsCode}(i) = 9$ or $\mathrm{InsCode}(i) = 10$,

(iv) there exists an integer location a and there exists a finite sequence location f such that $i = a{:=}\mathrm{len} f$ or $i = f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle$ but $\mathrm{UsedIntLoc}(i) = \{a\}$ if $\mathrm{InsCode}(i) = 11$ or $\mathrm{InsCode}(i) = 12$,

(v) $\mathrm{UsedIntLoc}(i) = \emptyset$, otherwise.

Next we state several propositions:

(17) $\mathrm{UsedIntLoc}(\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}) = \emptyset$.

(18) If $i = a{:=}b$ or $i = \mathrm{AddTo}(a, b)$ or $i = \mathrm{SubFrom}(a, b)$ or $i = \mathrm{MultBy}(a, b)$ or $i = \mathrm{Divide}(a, b)$, then $\mathrm{UsedIntLoc}(i) = \{a, b\}$.

(19) $\mathrm{UsedIntLoc}(\mathrm{goto}\ l) = \emptyset$.

(20) If $i = \mathbf{if}\ a = 0\ \mathbf{goto}\ l$ or $i = \mathbf{if}\ a > 0\ \mathbf{goto}\ l$, then $\mathrm{UsedIntLoc}(i) = \{a\}$.

(21) If $i = b{:=}f_a$ or $i = f_a{:=}b$, then $\mathrm{UsedIntLoc}(i) = \{a, b\}$.

(22) If $i = a{:=}\mathrm{len} f$ or $i = f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle$, then $\mathrm{UsedIntLoc}(i) = \{a\}$.

Let p be a programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. The functor $\mathrm{UsedIntLoc}(p)$ yields a subset of Int-Locations and is defined by the condition (Def. 2).

(Def. 2) There exists a function U_1 from the instructions of $\mathbf{SCM}_{\mathrm{FSA}}$ into $\mathrm{Fin}\,\text{Int-Locations}$ such that for every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $U_1(i) = \mathrm{UsedIntLoc}(i)$ and $\mathrm{UsedIntLoc}(p) = \mathrm{Union}(U_1 \cdot p)$.

Let p be a programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. Note that $\mathrm{UsedIntLoc}(p)$ is finite.

We follow the rules: p, r denote programmed finite partial states of $\mathbf{SCM}_{\mathrm{FSA}}$, I, J denote macro instructions, and k, m, n denote natural numbers.

Next we state a number of propositions:

(23) If $i \in \mathrm{rng}\, p$, then $\mathrm{UsedIntLoc}(i) \subseteq \mathrm{UsedIntLoc}(p)$.

(24) $\mathrm{UsedIntLoc}(p {+\cdot} r) \subseteq \mathrm{UsedIntLoc}(p) \cup \mathrm{UsedIntLoc}(r)$.

(25) If $\mathrm{dom}\, p$ misses $\mathrm{dom}\, r$, then $\mathrm{UsedIntLoc}(p {+\cdot} r) = \mathrm{UsedIntLoc}(p) \cup \mathrm{UsedIntLoc}(r)$.

(26) $\mathrm{UsedIntLoc}(p) = \mathrm{UsedIntLoc}(\mathrm{Shift}(p, k))$.

(27) $\mathrm{UsedIntLoc}(i) = \mathrm{UsedIntLoc}(\mathrm{IncAddr}(i, k))$.

(28) $\mathrm{UsedIntLoc}(p) = \mathrm{UsedIntLoc}(\mathrm{IncAddr}(p, k))$.

(29) $\mathrm{UsedIntLoc}(I) = \mathrm{UsedIntLoc}(\mathrm{ProgramPart}(\mathrm{Relocated}(I, k)))$.

(30) $\mathrm{UsedIntLoc}(I) = \mathrm{UsedIntLoc}(\mathrm{Directed}(I))$.

(31) $\mathrm{UsedIntLoc}(I;J) = \mathrm{UsedIntLoc}(I) \cup \mathrm{UsedIntLoc}(J)$.

(32) $\mathrm{UsedIntLoc}(\mathrm{Macro}(i)) = \mathrm{UsedIntLoc}(i)$.

(33) $\mathrm{UsedIntLoc}(i;J) = \mathrm{UsedIntLoc}(i) \cup \mathrm{UsedIntLoc}(J)$.

(34) $\mathrm{UsedIntLoc}(I;j) = \mathrm{UsedIntLoc}(I) \cup \mathrm{UsedIntLoc}(j)$.

(35) $\mathrm{UsedIntLoc}(i;j) = \mathrm{UsedIntLoc}(i) \cup \mathrm{UsedIntLoc}(j)$.

4. Finite sequence locations used in macros

Let i be an instruction of $\mathbf{SCM}_{\rm FSA}$. The functor UsedInt* Loc(i) yielding an element of Fin FinSeq-Locations is defined by:

(Def. 3) (i) There exist integer locations a, b and there exists a finite sequence location f such that $i = b{:=}f_a$ or $i = f_a{:=}b$ but UsedInt* Loc$(i) = \{f\}$ if InsCode$(i) = 9$ or InsCode$(i) = 10$,

(ii) there exists an integer location a and there exists a finite sequence location f such that $i = a{:=}{\rm len}f$ or $i = f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle$ but UsedInt* Loc$(i) = \{f\}$ if InsCode$(i) = 11$ or InsCode$(i) = 12$,

(iii) UsedInt* Loc$(i) = \emptyset$, otherwise.

One can prove the following propositions:

(36) If $i = \mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$ or $i = a{:=}b$ or $i = {\rm AddTo}(a,b)$ or $i = {\rm SubFrom}(a,b)$ or $i = {\rm MultBy}(a,b)$ or $i = {\rm Divide}(a,b)$ or $i = {\rm goto}\ l$ or $i = \mathbf{if}\ a = 0\ \mathbf{goto}\ l$ or $i = \mathbf{if}\ a > 0\ \mathbf{goto}\ l$, then UsedInt* Loc$(i) = \emptyset$.

(37) If $i = b{:=}f_a$ or $i = f_a{:=}b$, then UsedInt* Loc$(i) = \{f\}$.

(38) If $i = a{:=}{\rm len}f$ or $i = f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle$, then UsedInt* Loc$(i) = \{f\}$.

Let p be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$. The functor UsedInt* Loc(p) yields a subset of FinSeq-Locations and is defined by the condition (Def. 4).

(Def. 4) There exists a function U_1 from the instructions of $\mathbf{SCM}_{\rm FSA}$ into Fin FinSeq-Locations such that for every instruction i of $\mathbf{SCM}_{\rm FSA}$ holds $U_1(i) =$ UsedInt* Loc(i) and UsedInt* Loc$(p) = {\rm Union}(U_1 \cdot p)$.

Let p be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$. Note that UsedInt* Loc(p) is finite.

The following propositions are true:

(39) If $i \in {\rm rng}\, p$, then UsedInt* Loc$(i) \subseteq$ UsedInt* Loc(p).

(40) UsedInt* Loc$(p{+\cdot}r) \subseteq$ UsedInt* Loc$(p) \cup$ UsedInt* Loc(r).

(41) If ${\rm dom}\, p$ misses ${\rm dom}\, r$, then UsedInt* Loc$(p{+\cdot}r) =$ UsedInt* Loc$(p) \cup$ UsedInt* Loc(r).

(42) UsedInt* Loc$(p) =$ UsedInt* Loc$({\rm Shift}(p,k))$.

(43) UsedInt* Loc$(i) =$ UsedInt* Loc$({\rm IncAddr}(i,k))$.

(44) UsedInt* Loc$(p) =$ UsedInt* Loc$({\rm IncAddr}(p,k))$.

(45) UsedInt* Loc$(I) =$ UsedInt* Loc$({\rm ProgramPart}({\rm Relocated}(I,k)))$.

(46) UsedInt* Loc$(I) =$ UsedInt* Loc$({\rm Directed}(I))$.

(47) UsedInt* Loc$(I;J) =$ UsedInt* Loc$(I) \cup$ UsedInt* Loc(J).

(48) UsedInt* Loc$({\rm Macro}(i)) =$ UsedInt* Loc(i).

(49) UsedInt* Loc$(i;J) =$ UsedInt* Loc$(i) \cup$ UsedInt* Loc(J).

(50) UsedInt* Loc$(I;j) =$ UsedInt* Loc$(I) \cup$ UsedInt* Loc(j).

(51) $\mathrm{UsedInt}^* \mathrm{Loc}(i;j) = \mathrm{UsedInt}^* \mathrm{Loc}(i) \cup \mathrm{UsedInt}^* \mathrm{Loc}(j)$.

5. Choosing an integer location not used in a program

Let I_1 be an integer location. We say that I_1 is read-only if and only if:

(Def. 5) $I_1 = \mathrm{intloc}(0)$.

We introduce I_1 is read-write as an antonym of I_1 is read-only.

Let us observe that intloc(0) is read-only.

One can check that there exists an integer location which is read-write.

In the sequel L will be a finite subset of Int-Locations.

Let L be a finite subset of Int-Locations. The functor FirstNotIn(L) yields an integer location and is defined by:

(Def. 6) There exists a non empty subset s_1 of $\mathbb{N}$ such that $\mathrm{FirstNotIn}(L) = \mathrm{intloc}(\min s_1)$ and $s_1 = \{k : k$ ranges over natural numbers, $\mathrm{intloc}(k) \notin L\}$.

Next we state two propositions:

(52) $\mathrm{FirstNotIn}(L) \notin L$.

(53) If $\mathrm{FirstNotIn}(L) = \mathrm{intloc}(m)$ and $\mathrm{intloc}(n) \notin L$, then $m \leq n$.

Let p be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$. The functor FirstNotUsed(p) yields an integer location and is defined by:

(Def. 7) There exists a finite subset s_2 of Int-Locations such that $s_2 = \mathrm{UsedIntLoc}(p) \cup \{\mathrm{intloc}(0)\}$ and $\mathrm{FirstNotUsed}(p) = \mathrm{FirstNotIn}(s_2)$.

Let p be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$. Observe that FirstNotUsed(p) is read-write.

We now state several propositions:

(54) $\mathrm{FirstNotUsed}(p) \notin \mathrm{UsedIntLoc}(p)$.

(55) If $a{:=}b \in \mathrm{rng}\, p$ or $\mathrm{AddTo}(a,b) \in \mathrm{rng}\, p$ or $\mathrm{SubFrom}(a,b) \in \mathrm{rng}\, p$ or $\mathrm{MultBy}(a,b) \in \mathrm{rng}\, p$ or $\mathrm{Divide}(a,b) \in \mathrm{rng}\, p$, then $\mathrm{FirstNotUsed}(p) \neq a$ and $\mathrm{FirstNotUsed}(p) \neq b$.

(56) If $\mathbf{if}\ a = 0\ \mathbf{goto}\ l \in \mathrm{rng}\, p$ or $\mathbf{if}\ a > 0\ \mathbf{goto}\ l \in \mathrm{rng}\, p$, then $\mathrm{FirstNotUsed}(p) \neq a$.

(57) If $b{:=}f_a \in \mathrm{rng}\, p$ or $f_a{:=}b \in \mathrm{rng}\, p$, then $\mathrm{FirstNotUsed}(p) \neq a$ and $\mathrm{FirstNotUsed}(p) \neq b$.

(58) If $a{:=}\mathrm{len} f \in \mathrm{rng}\, p$ or $f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle \in \mathrm{rng}\, p$, then $\mathrm{FirstNotUsed}(p) \neq a$.

6. Choosing a finite sequence location not used in a program

In the sequel L is a finite subset of FinSeq-Locations.

Let L be a finite subset of FinSeq-Locations. The functor $\mathrm{First}^* \mathrm{NotIn}(L)$ yielding a finite sequence location is defined by:

(Def. 8) There exists a non empty subset s_1 of $\mathbb{N}$ such that $\mathrm{First}^* \mathrm{NotIn}(L) = \mathrm{fsloc}(\min s_1)$ and $s_1 = \{k : k$ ranges over natural numbers, $\mathrm{fsloc}(k) \notin L\}$.

We now state two propositions:

(59) $\mathrm{First}^* \mathrm{NotIn}(L) \notin L$.

(60) If $\mathrm{First}^* \mathrm{NotIn}(L) = \mathrm{fsloc}(m)$ and $\mathrm{fsloc}(n) \notin L$, then $m \leq n$.

Let p be a programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. The functor $\mathrm{First}^* \mathrm{NotUsed}(p)$ yields a finite sequence location and is defined by:

(Def. 9) There exists a finite subset s_2 of FinSeq-Locations such that $s_2 = \mathrm{UsedInt}^* \mathrm{Loc}(p)$ and $\mathrm{First}^* \mathrm{NotUsed}(p) = \mathrm{First}^* \mathrm{NotIn}(s_2)$.

One can prove the following propositions:

(61) $\mathrm{First}^* \mathrm{NotUsed}(p) \notin \mathrm{UsedInt}^* \mathrm{Loc}(p)$.

(62) If $b{:=}f_a \in \mathrm{rng}\, p$ or $f_a{:=}b \in \mathrm{rng}\, p$, then $\mathrm{First}^* \mathrm{NotUsed}(p) \neq f$.

(63) If $a{:=}\mathrm{len} f \in \mathrm{rng}\, p$ or $f{:=}\langle \underbrace{0, \ldots, 0}_{a} \rangle \in \mathrm{rng}\, p$, then $\mathrm{First}^* \mathrm{NotUsed}(p) \neq f$.

7. Semantics

In the sequel s, t will be states of $\mathbf{SCM}_{\mathrm{FSA}}$.

We now state a number of propositions:

(64) $\mathrm{dom}\, I \cap \mathrm{dom}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(n)) = \emptyset$.

(65) $\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}} \in \mathrm{dom}(I {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(n)))$.

(66) $(I {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(n)))(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}) = \mathrm{insloc}(n)$.

(67) If $I {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(n)) \subseteq s$, then $\mathbf{IC}_s = \mathrm{insloc}(n)$.

(68) If $c \notin \mathrm{UsedIntLoc}(i)$, then $(\mathrm{Exec}(i, s))(c) = s(c)$.

(69) If $I {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s$ and for every m such that $m < n$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(m)} \in \mathrm{dom}\, I$ and $a \notin \mathrm{UsedIntLoc}(I)$, then $(\mathrm{Computation}(s))(n)(a) = s(a)$.

(70) If $f \notin \mathrm{UsedInt}^* \mathrm{Loc}(i)$, then $(\mathrm{Exec}(i, s))(f) = s(f)$.

(71) If $I {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s$ and for every m such that $m < n$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(m)} \in \mathrm{dom}\, I$ and $f \notin \mathrm{UsedInt}^* \mathrm{Loc}(I)$, then $(\mathrm{Computation}(s))(n)(f) = s(f)$.

(72) If $s \restriction \mathrm{UsedIntLoc}(i) = t \restriction \mathrm{UsedIntLoc}(i)$ and $s \restriction \mathrm{UsedInt}^* \mathrm{Loc}(i) = t \restriction \mathrm{UsedInt}^* \mathrm{Loc}(i)$ and $\mathbf{IC}_s = \mathbf{IC}_t$, then $\mathbf{IC}_{\mathrm{Exec}(i,s)} = \mathbf{IC}_{\mathrm{Exec}(i,t)}$ and $\mathrm{Exec}(i, s) \restriction \mathrm{UsedIntLoc}(i) = \mathrm{Exec}(i, t) \restriction \mathrm{UsedIntLoc}(i)$ and $\mathrm{Exec}(i, s) \restriction \mathrm{UsedInt}^* \mathrm{Loc}(i) = \mathrm{Exec}(i, t) \restriction \mathrm{UsedInt}^* \mathrm{Loc}(i)$.

(73) Suppose $I +\cdot$ Start-At(insloc(0)) $\subseteq s$ and $I +\cdot$ Start-At(insloc(0)) $\subseteq t$ and $s \upharpoonright \mathrm{UsedIntLoc}(I) = t \upharpoonright \mathrm{UsedIntLoc}(I)$ and $s \upharpoonright \mathrm{UsedInt}^* \mathrm{Loc}(I) = t \upharpoonright \mathrm{UsedInt}^* \mathrm{Loc}(I)$ and for every m such that $m < n$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(m)} \in \mathrm{dom}\, I$. Then

(i) for every m such that $m < n$ holds $\mathbf{IC}_{(\mathrm{Computation}(t))(m)} \in \mathrm{dom}\, I$, and

(ii) for every m such that $m \leq n$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(m)} = \mathbf{IC}_{(\mathrm{Computation}(t))(m)}$ and for every a such that $a \in \mathrm{UsedIntLoc}(I)$ holds $(\mathrm{Computation}(s))(m)(a) = (\mathrm{Computation}(t))(m)(a)$ and for every f such that $f \in \mathrm{UsedInt}^* \mathrm{Loc}(I)$ holds $(\mathrm{Computation}(s))(m)(f) = (\mathrm{Computation}(t))(m)(f)$.

References

[1] Grzegorz Bancerek. Cardinal numbers. *Formalized Mathematics*, 1(**2**):377–382, 1990.
[2] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(**1**):41–46, 1990.
[3] Grzegorz Bancerek. König's theorem. *Formalized Mathematics*, 1(**3**):589–593, 1990.
[4] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 1990.
[5] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Formalized Mathematics*, 5(**4**):485–492, 1996.
[6] Czesław Byliński. A classical first order language. *Formalized Mathematics*, 1(**4**):669–676, 1990.
[7] Czesław Byliński. Finite sequences and tuples of elements of a non-empty sets. *Formalized Mathematics*, 1(**3**):529–536, 1990.
[8] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.
[9] Czesław Byliński. Functions from a set to a set. *Formalized Mathematics*, 1(**1**):153–164, 1990.
[10] Czesław Byliński. The modification of a function by a function and the iteration of the composition of a function. *Formalized Mathematics*, 1(**3**):521–527, 1990.
[11] Czesław Byliński. Products and coproducts in categories. *Formalized Mathematics*, 2(**5**):701–709, 1991.
[12] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(**1**):165–167, 1990.
[13] Agata Darmochwał and Andrzej Trybulec. Similarity of formulae. *Formalized Mathematics*, 2(**5**):635–642, 1991.
[14] Krzysztof Hryniewiecki. Basic properties of real numbers. *Formalized Mathematics*, 1(**1**):35–40, 1990.
[15] Rafał Kwiatek and Grzegorz Zwara. The divisibility of integers and integer relative primes. *Formalized Mathematics*, 1(**5**):829–832, 1990.
[16] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Formalized Mathematics*, 3(**2**):151–160, 1992.
[17] Andrzej Nędzusiak. σ-fields and probability. *Formalized Mathematics*, 1(**2**):401–407, 1990.
[18] Yasushi Tanaka. On the decomposition of the states of SCM. *Formalized Mathematics*, 5(**1**):1–8, 1996.
[19] Andrzej Trybulec. Enumerated sets. *Formalized Mathematics*, 1(**1**):25–34, 1990.
[20] Andrzej Trybulec. Semilattice operations on finite subsets. *Formalized Mathematics*, 1(**2**):369–376, 1990.
[21] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[22] Andrzej Trybulec and Agata Darmochwał. Boolean domains. *Formalized Mathematics*, 1(**1**):187–190, 1990.

[23] Andrzej Trybulec and Yatsuka Nakamura. Modifying addresses of instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. *Formalized Mathematics*, 5(**4**):571–576, 1996.
[24] Andrzej Trybulec and Yatsuka Nakamura. Relocability for $\mathbf{SCM}_{\mathrm{FSA}}$. *Formalized Mathematics*, 5(**4**):583–586, 1996.
[25] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Formalized Mathematics*, 4(**1**):51–56, 1993.
[26] Andrzej Trybulec, Yatsuka Nakamura, and Noriko Asamoto. On the compositions of macro instructions. Part I. *Formalized Mathematics*, 6(**1**):21–27, 1997.
[27] Andrzej Trybulec, Yatsuka Nakamura, and Piotr Rudnicki. The $\mathbf{SCM}_{\mathrm{FSA}}$ computer. *Formalized Mathematics*, 5(**4**):519–528, 1996.
[28] Michał J. Trybulec. Integers. *Formalized Mathematics*, 1(**3**):501–505, 1990.
[29] Wojciech A. Trybulec. Pigeon hole principle. *Formalized Mathematics*, 1(**3**):575–579, 1990.
[30] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.
[31] Zinaida Trybulec and Halina Święczkowska. Boolean properties of sets. *Formalized Mathematics*, 1(**1**):17–23, 1990.
[32] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.

Received July 18, 1996