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

On the Compositions of Macro Instructions. Part I

Andrzej Trybulec
Warsaw University
Białystok

Yatsuka Nakamura
Shinshu University
Nagano

Noriko Asamoto
Ochanomizu University
Tokyo

MML Identifier: SCMFSA6A.

The notation and terminology used here are introduced in the following papers: [21], [28], [14], [2], [26], [17], [29], [8], [9], [3], [7], [27], [11], [1], [19], [6], [12], [13], [10], [20], [15], [16], [24], [4], [18], [5], [25], [22], and [23].

1. Preliminaries

One can prove the following propositions:

(1) For all functions f, g and for all sets x, y such that $g \subseteq f$ and $x \notin \operatorname{dom} g$ holds $g \subseteq f +\cdot (x, y)$.

(2) For all functions f, g and for every set A such that $f \restriction A = g \restriction A$ and f and g are equal outside A holds $f = g$.

(3) For every function f and for all sets a, b, A such that $a \in A$ holds f and $f +\cdot (a, b)$ are equal outside A.

(4) For every function f and for all sets a, b, A holds $a \in A$ or $(f +\cdot (a, b)) \restriction A = f \restriction A$.

(5) For all functions f, g and for all sets a, b, A such that $f \restriction A = g \restriction A$ holds $(f +\cdot (a, b)) \restriction A = (g +\cdot (a, b)) \restriction A$.

(6) For all functions f, g, h such that $f \subseteq h$ and $g \subseteq h$ holds $f +\cdot g \subseteq h$.

(7) For arbitrary a, b and for every function f holds $a \dot{\longmapsto} b \subseteq f$ iff $a \in \operatorname{dom} f$ and $f(a) = b$.

(8) For every function f and for every set A holds $\operatorname{dom}(f \restriction (\operatorname{dom} f \setminus A)) = \operatorname{dom} f \setminus A$.

ISSN 1426–2630

(9) Let f, g be functions and let D be a set. Suppose $D \subseteq \operatorname{dom} f$ and $D \subseteq \operatorname{dom} g$. Then $f \upharpoonright D = g \upharpoonright D$ if and only if for arbitrary x such that $x \in D$ holds $f(x) = g(x)$.

(10) For every function f and for every set D holds $f \upharpoonright D = f \upharpoonright (\operatorname{dom} f \cap D)$.

(11) Let f, g, h be functions and let A be a set. Suppose f and g are equal outside A. Then $f+\cdot h$ and $g+\cdot h$ are equal outside A.

(12) Let f, g, h be functions and let A be a set. Suppose f and g are equal outside A. Then $h+\cdot f$ and $h+\cdot g$ are equal outside A.

(13) For all functions f, g, h holds $f+\cdot h = g+\cdot h$ iff f and g are equal outside $\operatorname{dom} h$.

2. Macroinstructions

A macro instruction is an initial programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$.

We follow a convention: m, n denote natural numbers, i, j, k denote instructions of $\mathbf{SCM}_{\mathrm{FSA}}$, and I, J, K denote macro instructions.

Let I be a programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. The functor $\mathrm{Directed}(I)$ yields a programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$ and is defined by:

(Def. 1) $\mathrm{Directed}(I) = (\mathrm{id}_{(\text{the instructions of } \mathbf{SCM}_{\mathrm{FSA}})} +\cdot (\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}} \dot{\longmapsto} \mathrm{goto}\ \mathrm{insloc}(\operatorname{card} I))) \cdot I$.

The following proposition is true

(14) $\operatorname{dom} \mathrm{Directed}(I) = \operatorname{dom} I$.

Let I be a macro instruction. Note that $\mathrm{Directed}(I)$ is initial.

Let us consider i. The functor $\mathrm{Macro}(i)$ yields a macro instruction and is defined by:

(Def. 2) $\mathrm{Macro}(i) = [\mathrm{insloc}(0) \longmapsto i, \mathrm{insloc}(1) \longmapsto \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}]$.

Let us consider i. One can check that $\mathrm{Macro}(i)$ is non empty.

We now state the proposition

(15) For every macro instruction P and for every n holds $n < \operatorname{card} P$ iff $\mathrm{insloc}(n) \in \operatorname{dom} P$.

Let I be an initial finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. Observe that $\mathrm{ProgramPart}(I)$ is initial.

One can prove the following propositions:

(16) $\operatorname{dom} I$ misses $\operatorname{dom} \mathrm{ProgramPart}(\mathrm{Relocated}(J, \operatorname{card} I))$.

(17) For every programmed finite partial state I of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\operatorname{card} \mathrm{ProgramPart}(\mathrm{Relocated}(I, m)) = \operatorname{card} I$.

(18) $\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}} \notin \operatorname{rng} \mathrm{Directed}(I)$.

(19) $\mathrm{ProgramPart}(\mathrm{Relocated}(\mathrm{Directed}(I), m)) = (\mathrm{id}_{(\text{the instructions of } \mathbf{SCM}_{\mathrm{FSA}})} +\cdot (\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}} \dot{\longmapsto} \mathrm{goto}\ \mathrm{insloc}(m+\operatorname{card} I))) \cdot \mathrm{ProgramPart}(\mathrm{Relocated}(I, m))$.

(20) For all finite partial states I, J of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{ProgramPart}(I+\cdot J) = \mathrm{ProgramPart}(I) +\cdot \mathrm{ProgramPart}(J)$.

(21) For all finite partial states I, J of $\mathbf{SCM}_{\mathrm{FSA}}$ holds ProgramPart (Relocated($I{+\cdot}J, n$)) = ProgramPart(Relocated(I, n)) $+\cdot$ ProgramPart(Relocated(J, n)).

(22) ProgramPart(Relocated(ProgramPart(Relocated(I, m)), n)) = ProgramPart(Relocated($I, m + n$)).

In the sequel s, s_1, s_2 denote states of $\mathbf{SCM}_{\mathrm{FSA}}$.

Let us consider I. The functor Initialized(I) yields a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$ and is defined by:

(Def. 3) Initialized(I) = $I{+\cdot}$(intloc(0)$\longmapsto$1)$+\cdot$ Start-At(insloc(0)).

Next we state a number of propositions:

(23) InsCode(i) $\in \{0, 6, 7, 8\}$ or (Exec(i, s))($\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}$) = Next($\mathbf{IC}_s$).

(24) $\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}} \in$ dom Initialized(I).

(25) $\mathbf{IC}_{\mathrm{Initialized}(I)}$ = insloc(0).

(26) $I \subseteq$ Initialized(I).

(27) s and $s{+\cdot}I$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(28) Let s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose $\mathbf{IC}_{(s_1)} = \mathbf{IC}_{(s_2)}$ and for every integer location a holds $s_1(a) = s_2(a)$ and for every finite sequence location f holds $s_1(f) = s_2(f)$. Then s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(29) If s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$, then $\mathbf{IC}_{(s_1)} = \mathbf{IC}_{(s_2)}$.

(30) Suppose s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$. Let a be an integer location. Then $s_1(a) = s_2(a)$.

(31) Suppose s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$. Let f be a finite sequence location. Then $s_1(f) = s_2(f)$.

(32) Suppose s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$. Then Exec($i, s_1$) and Exec($i, s_2$) are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(33) Initialized(I) $\restriction$ (the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$) = I.

The scheme *SCMFSAEx* deals with a unary functor $\mathcal{F}$ yielding an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$, a unary functor $\mathcal{G}$ yielding an integer, a unary functor $\mathcal{H}$ yielding a finite sequence of elements of $\mathbb{Z}$, and an instruction-location $\mathcal{A}$ of $\mathbf{SCM}_{\mathrm{FSA}}$, and states that:

There exists a state S of $\mathbf{SCM}_{\mathrm{FSA}}$ such that $\mathbf{IC}_S = \mathcal{A}$ and for every natural number i holds S(insloc(i)) = $\mathcal{F}(i)$ and S(intloc(i)) = $\mathcal{G}(i)$ and S(fsloc(i)) = $\mathcal{H}(i)$

for all values of the parameters.

One can prove the following propositions:

(34) For every state s of $\mathbf{SCM}_{\mathrm{FSA}}$ holds dom s = Int-Locations $\cup$ FinSeq-Locations $\cup \{\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}\} \cup$ the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(35) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and let x be arbitrary. Suppose $x \in$ dom s. Then

(i) x is an integer location or a finite sequence location, or

(ii) $x = \mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}$, or

(iii) x is an instruction-location of $\mathbf{SCM}_{\mathrm{FSA}}$.

(36) Let s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$. Then for every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $s_1(l) = s_2(l)$ if and only if $s_1 \restriction$ (the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}) = s_2 \restriction$ (the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$).

(37) For every instruction-location i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $i \notin \mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}$ and $\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}} \notin \mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}$.

(38) Let s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$. Then for every integer location a holds $s_1(a) = s_2(a)$ and for every finite sequence location f holds $s_1(f) = s_2(f)$ if and only if $s_1 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = s_2 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

(39) Let s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.
Then $s_1 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = s_2 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

(40) For all states s, s_3 of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every set A holds $(s_3 {+}\!\cdot s \restriction A) \restriction A = s \restriction A$.

(41) Let s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$, and let n be a natural number, and let i be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose $\mathbf{IC}_{(s_1)} + n = \mathbf{IC}_{(s_2)}$ and $s_1 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = s_2 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$. Then $\mathbf{IC}_{\mathrm{Exec}(i,s_1)} + n = \mathbf{IC}_{\mathrm{Exec}(\mathrm{IncAddr}(i,n),s_2)}$ and $\mathrm{Exec}(i, s_1) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = \mathrm{Exec}(\mathrm{IncAddr}(i, n), s_2) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

(42) For all macro instructions I, J holds I and J are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(43) For every macro instruction I holds $\mathrm{dom}\,\mathrm{Initialized}(I) = \mathrm{dom}\, I \cup \{\mathrm{intloc}(0)\} \cup \{\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}\}$.

(44) For every macro instruction I and for arbitrary x such that $x \in \mathrm{dom}\,\mathrm{Initialized}(I)$ holds $x \in \mathrm{dom}\, I$ or $x = \mathrm{intloc}(0)$ or $x = \mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(45) For every macro instruction I holds $\mathrm{intloc}(0) \in \mathrm{dom}\,\mathrm{Initialized}(I)$.

(46) For every macro instruction I holds $(\mathrm{Initialized}(I))(\mathrm{intloc}(0)) = 1$ and $(\mathrm{Initialized}(I))(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}) = \mathrm{insloc}(0)$.

(47) For every macro instruction I holds $\mathrm{intloc}(0) \notin \mathrm{dom}\, I$ and $\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}} \notin \mathrm{dom}\, I$.

(48) For every macro instruction I and for every integer location a such that $a \neq \mathrm{intloc}(0)$ holds $a \notin \mathrm{dom}\,\mathrm{Initialized}(I)$.

(49) For every macro instruction I and for every finite sequence location f holds $f \notin \mathrm{dom}\,\mathrm{Initialized}(I)$.

(50) For every macro instruction I and for arbitrary x such that $x \in \mathrm{dom}\, I$ holds $I(x) = (\mathrm{Initialized}(I))(x)$.

(51) For all macro instructions I, J and for every state s of $\mathbf{SCM}_{\rm FSA}$ such that $\mathrm{Initialized}(J) \subseteq s$ holds $s +\cdot \mathrm{Initialized}(I) = s +\cdot I$.

(52) For all macro instructions I, J and for every state s of $\mathbf{SCM}_{\rm FSA}$ such that $\mathrm{Initialized}(J) \subseteq s$ holds $\mathrm{Initialized}(I) \subseteq s +\cdot I$.

(53) Let I, J be macro instructions and let s be a state of $\mathbf{SCM}_{\rm FSA}$. Then $s +\cdot \mathrm{Initialized}(I)$ and $s +\cdot \mathrm{Initialized}(J)$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$.

3. The composition of macroinstructions

Let I, J be macro instructions. The functor $I;J$ yields a macro instruction and is defined by:

(Def. 4) $I;J = \mathrm{Directed}(I) +\cdot \mathrm{ProgramPart}(\mathrm{Relocated}(J, \mathrm{card}\, I))$.

Let I, J be macro instructions. Note that $I;J$ is initial.

Next we state several propositions:

(54) Let I, J be macro instructions and let l be an instruction-location of $\mathbf{SCM}_{\rm FSA}$. If $l \in \mathrm{dom}\, I$ and $I(l) \neq \mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$, then $(I;J)(l) = I(l)$.

(55) For all macro instructions I, J holds $\mathrm{Directed}(I) \subseteq I;J$.

(56) For all macro instructions I, J holds $\mathrm{dom}\, I \subseteq \mathrm{dom}(I;J)$.

(57) For all macro instructions I, J holds $I +\cdot (I;J) = I;J$.

(58) For all macro instructions I, J holds $\mathrm{Initialized}(I) +\cdot (I;J) = \mathrm{Initialized}(I;J)$.

4. The compostion of instruction and macroinstructions

Let us consider i, J. The functor $i;J$ yielding a macro instruction is defined as follows:

(Def. 5) $i;J = \mathrm{Macro}(i);J$.

Let us consider I, j. The functor $I;j$ yields a macro instruction and is defined by:

(Def. 6) $I;j = I; \mathrm{Macro}(j)$.

Let us consider i, j. The functor $i;j$ yields a macro instruction and is defined by:

(Def. 7) $i;j = \mathrm{Macro}(i); \mathrm{Macro}(j)$.

The following propositions are true:

(59) $i;j = \mathrm{Macro}(i);j$.

(60) $i;j = i; \mathrm{Macro}(j)$.

(61) $\mathrm{card}(I;J) = \mathrm{card}\, I + \mathrm{card}\, J$.

(62) $(I;J);K = I;(J;K)$.

(63) $(I;J);k = I;(J;k)$.

(64) $(I;j);K = I;(j;K)$.

(65) $(I;j);k = I;(j;k)$.

(66) $(i;J);K = i;(J;K)$.

(67) $(i;J);k = i;(J;k)$.

(68) $(i;j);K = i;(j;K)$.

(69) $(i;j);k = i;(j;k)$.

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 and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 1990.
[4] Grzegorz Bancerek and Piotr Rudnicki. Development of terminology for **SCM**. *Formalized Mathematics*, 4(**1**):61–67, 1993.
[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. Partial functions. *Formalized Mathematics*, 1(**2**):357–367, 1990.
[12] Czesław Byliński. Products and coproducts in categories. *Formalized Mathematics*, 2(**5**):701–709, 1991.
[13] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(**1**):165–167, 1990.
[14] Krzysztof Hryniewiecki. Basic properties of real numbers. *Formalized Mathematics*, 1(**1**):35–40, 1990.
[15] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Formalized Mathematics*, 3(**2**):151–160, 1992.
[16] Yatsuka Nakamura and Andrzej Trybulec. On a mathematical model of programs. *Formalized Mathematics*, 3(**2**):241–250, 1992.
[17] Jan Popiołek. Some properties of functions modul and signum. *Formalized Mathematics*, 1(**2**):263–264, 1990.
[18] Yasushi Tanaka. On the decomposition of the states of SCM. *Formalized Mathematics*, 5(**1**):1–8, 1996.
[19] Andrzej Trybulec. Binary operations applied to functions. *Formalized Mathematics*, 1(**2**):329–334, 1990.
[20] Andrzej Trybulec. Enumerated sets. *Formalized Mathematics*, 1(**1**):25–34, 1990.
[21] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[22] Andrzej Trybulec and Yatsuka Nakamura. Modifying addresses of instructions of $\mathbf{SCM}_{\rm FSA}$. *Formalized Mathematics*, 5(**4**):571–576, 1996.
[23] Andrzej Trybulec and Yatsuka Nakamura. Relocability for $\mathbf{SCM}_{\rm FSA}$. *Formalized Mathematics*, 5(**4**):583–586, 1996.
[24] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Formalized Mathematics*, 4(**1**):51–56, 1993.

[25] Andrzej Trybulec, Yatsuka Nakamura, and Piotr Rudnicki. The $\mathbf{SCM}_{\mathrm{FSA}}$ computer. *Formalized Mathematics*, 5(**4**):519–528, 1996.
[26] Michał J. Trybulec. Integers. *Formalized Mathematics*, 1(**3**):501–505, 1990.
[27] Wojciech A. Trybulec. Pigeon hole principle. *Formalized Mathematics*, 1(**3**):575–579, 1990.
[28] Zinaida Trybulec and Halina Święczkowska. Boolean properties of sets. *Formalized Mathematics*, 1(**1**):17–23, 1990.
[29] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.

Received June 20, 1996