JOURNAL OF FORMALIZED MATHEMATICS
Volume 8, Released 1996, Published 2003
Inst. of Computer Science, Univ. of 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.

WWW: http://mizar.org/JFM/Vol8/scmfsa6a.html

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

1. Preliminaries

The following propositions are true:

(1) For all functions f, g and for all sets x, y such that $g \subseteq f$ and $x \notin \mathrm{dom}\, g$ 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 all sets a, b and for every function f holds $a \dot{\longmapsto} b \subseteq f$ iff $a \in \mathrm{dom}\, f$ and $f(a) = b$.

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

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

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

(11) Let f, g, h be functions and 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 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 $\mathrm{dom}\, h$.

2. MACROINSTRUCTIONS

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

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

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

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

The following proposition is true

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

Let I be a macro instruction. Note that Directed(I) is initial.

Let us consider i. The functor Macro(i) yielding a macro instruction is defined as follows:

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

Let us consider i. Note that Macro(i) is non empty.

One can prove the following proposition

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

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

Next we state several propositions:

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

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

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

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

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

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

(22) $\mathrm{ProgramPart}(\mathrm{Relocated}(\mathrm{ProgramPart}(\mathrm{Relocated}(I,m)),n)) = \mathrm{ProgramPart}(\mathrm{Relocated}(I, m + n))$.

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

Let I be a finite partial state of $\mathbf{SCM}_{\rm FSA}$. The functor Initialized(I) yields a finite partial state of $\mathbf{SCM}_{\rm FSA}$ and is defined as follows:

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

One can prove the following propositions:

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

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

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

(26) $I \subseteq \mathrm{Initialized}(I)$.

(27) Let N be a set, A be an AMI over N, s be a state of A, and I be a programmed finite partial state of A. Then s and $s {+\cdot} I$ are equal outside the instruction locations of A.

(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) Let N be a set with non empty elements, S be a realistic IC-Ins-separated definite non empty non void AMI over N, and s_1, s_2 be states of S. Suppose s_1 and s_2 are equal outside the instruction locations of S. 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 $\mathrm{Exec}(i, s_1)$ and $\mathrm{Exec}(i, s_2)$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

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

The scheme *SCMFSAEx* deals with a unary functor $\mathcal{F}$ yielding an element of the instructions 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 $S(\mathrm{insloc}(i)) = \mathcal{F}(i)$ and $S(\mathrm{intloc}(i)) = \mathcal{G}(i)$ and $S(\mathrm{fsloc}(i)) = \mathcal{H}(i)$

for all values of the parameters.

One can prove the following propositions:

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

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

(i) x is an integer location and 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 $i \notin \text{Int-Locations} \cup \text{FinSeq-Locations}$ and $\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}} \notin \text{Int-Locations} \cup \text{FinSeq-Locations}$.

(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 $s_1 \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations}) = s_2 \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations})$.

(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 $s_1 \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations}) = s_2 \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations})$.

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

(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 every set 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 every set 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 $\mathbf{SCM}_{\mathrm{FSA}}$ 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 $\mathbf{SCM}_{\mathrm{FSA}}$ such that $\mathrm{Initialized}(J) \subseteq s$ holds $\mathrm{Initialized}(I) \subseteq s{+\cdot}I$.

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

3. The composition of macroinstructions

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

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

We now state several propositions:

(54) Let I, J be macro instructions and l be an instruction-location of $\mathbf{SCM}_{\mathrm{FSA}}$. If $l \in \mathrm{dom}\,I$ and $I(l) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$, 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 $i;J$ yields a macro instruction and is defined by:

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

Let us consider I, j. The functor $I;\,j$ yielding a macro instruction 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) $i;\,j = \mathrm{Macro}(i);\,\mathrm{Macro}(j)$.

We now state a number of propositions:

(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. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/card_1.html`.

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

[3] 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`.

[4] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/funct_7.html`.

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

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

[7] Czesław Byliński. A classical first order language. *Journal of Formalized Mathematics*, 2, 1990. `http://mizar.org/JFM/Vol2/cqc_lang.html`.

[8] Czesław Byliński. Finite sequences and tuples of elements of a non-empty sets. *Journal of Formalized Mathematics*, 2, 1990. `http://mizar.org/JFM/Vol2/finseq_2.html`.

[9] Czesław Byliński. The modification of a function by a function and the iteration of the composition of a function. *Journal of Formalized Mathematics*, 2, 1990. `http://mizar.org/JFM/Vol2/funct_4.html`.

[10] Agata Darmochwał. Finite sets. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/finset_1.html`.

[11] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Journal of Formalized Mathematics*, 4, 1992. `http://mizar.org/JFM/Vol4/ami_1.html`.

[12] Jan Popiołek. Some properties of functions modul and signum. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/absvalue.html`.

[13] Yasushi Tanaka. On the decomposition of the states of SCM. *Journal of Formalized Mathematics*, 5, 1993. `http://mizar.org/JFM/Vol5/ami_5.html`.

[14] Andrzej Trybulec. Enumerated sets. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/enumset1.html`.

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

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

[17] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Journal of Formalized Mathematics*, 5, 1993. `http://mizar.org/JFM/Vol5/ami_3.html`.

[18] Andrzej Trybulec and Yatsuka Nakamura. Modifying addresses of instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa_4.html`.

[19] Andrzej Trybulec and Yatsuka Nakamura. Relocability for $\mathbf{SCM}_{\rm FSA}$. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa_5.html`.

[20] Andrzej Trybulec, Yatsuka Nakamura, and Piotr Rudnicki. The $\mathbf{SCM}_{\rm FSA}$ computer. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa_2.html`.

[21] Michał J. Trybulec. Integers. *Journal of Formalized Mathematics*, 2, 1990. `http://mizar.org/JFM/Vol2/int_1.html`.

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

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

Received June 20, 1996

Published January 2, 2004