FORMALIZED MATHEMATICS
Volume 12, Number 2, 2004
University of Białystok

Relocability for SCM over Ring

Artur Korniłowicz[1]
University of Białystok

Yasunari Shidama
Shinshu University
Nagano

MML Identifier: SCMRING4.

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

1. On the Standard Computers

For simplicity, we use the following convention: i, j, k denote natural numbers, n denotes a natural number, N denotes a set with non empty elements, S denotes a standard IC-Ins-separated definite non empty non void AMI over N, l denotes an instruction-location of S, and f denotes a finite partial state of S.

Next we state the proposition

(1) $\mathbb{N} \approx$ the instruction locations of S.

Let us consider N, S. Observe that the instruction locations of S is infinite.

We now state the proposition

(2) $\mathrm{il}_S(i) + j = \mathrm{il}_S(i + j)$.

Let N be a set with non empty elements, let S be a standard IC-Ins-separated definite non empty non void AMI over N, let l_1 be an instruction-location of S, and let k be a natural number. The functor $l_1 -' k$ yields an instruction-location of S and is defined as follows:

(Def. 1) $l_1 -' k = \mathrm{il}_S(\mathrm{locnum}(l_1) -' k)$.

We now state a number of propositions:

(3) $l -' 0 = l$.

[1]The paper was written during the post-doctoral fellowship granted by Shinshu University, Japan.

© 2004 University of Białystok
ISSN 1426–2630

(4) $\mathrm{locnum}(l) -' k = \mathrm{locnum}(l -' k)$.

(5) $(l + k) -' k = l$.

(6) $\mathrm{il}_S(i) -' j = \mathrm{il}_S(i -' j)$.

(7) Let S be an IC-Ins-separated definite non empty non void AMI over N and p be a finite partial state of S. Then $\mathrm{dom}\,\mathrm{DataPart}(p) \subseteq$ (the carrier of S) $\setminus$ ($\{\mathbf{IC}_S\} \cup$ the instruction locations of S).

(8) Let S be an IC-Ins-separated definite realistic non empty non void AMI over N and p be a finite partial state of S. Then p is data-only if and only if $\mathrm{dom}\, p \subseteq$ (the carrier of S) $\setminus$ ($\{\mathbf{IC}_S\} \cup$ the instruction locations of S).

(9) For all instruction-locations l_2, l_3 of S holds Start-At$(l_2 + k)$ = Start-At$(l_3 + k)$ iff Start-At(l_2) = Start-At(l_3).

(10) For all instruction-locations l_2, l_3 of S such that Start-At(l_2) = Start-At(l_3) holds Start-At$(l_2 -' k)$ = Start-At$(l_3 -' k)$.

(11) If $l \in \mathrm{dom}\, f$, then $(\mathrm{Shift}(f, k))(l + k) = f(l)$.

(12) $\mathrm{dom}\,\mathrm{Shift}(f, k) = \{i_1 + k; i_1$ ranges over instruction-locations of S: $i_1 \in \mathrm{dom}\, f\}$.

(13) Let S be an Exec-preserving IC-Ins-separated definite realistic steady-programmed non empty non void AMI over N, s be a state of S, i be an instruction of S, and p be a programmed finite partial state of S. Then $\mathrm{Exec}(i, s{+\cdot}p) = \mathrm{Exec}(i, s){+\cdot}p$.

2. $\mathbf{SCM}(R)$

For simplicity, we follow the rules: R denotes a good ring, a, b denote Data-Locations of R, l_1 denotes an instruction-location of $\mathbf{SCM}(R)$, I denotes an instruction of $\mathbf{SCM}(R)$, p denotes a finite partial state of $\mathbf{SCM}(R)$, s, s_1, s_2 denote states of $\mathbf{SCM}(R)$, and q denotes a finite partial state of $\mathbf{SCM}$.

One can prove the following propositions:

(14) The carrier of $\mathbf{SCM}(R) = \{\mathbf{IC}_{\mathbf{SCM}(R)}\} \cup \mathrm{Data\text{-}Loc}_{\mathrm{SCM}} \cup \mathrm{Instr\text{-}Loc}_{\mathrm{SCM}}$.

(15) $\mathrm{ObjectKind}(l_1) = \mathrm{Instr}_{\mathrm{SCM}}(R)$.

(16) $\mathrm{dl}_R(n) = 2 \cdot n + 1$.

(17) $\mathrm{il}_{\mathbf{SCM}(R)}(k) = 2 \cdot k + 2$.

(18) For every Data-Location d_1 of R there exists a natural number i such that $d_1 = \mathrm{dl}_R(i)$.

(19) For all natural numbers i, j such that $i \neq j$ holds $\mathrm{dl}_R(i) \neq \mathrm{dl}_R(j)$.

(20) $a \neq l_1$.

(21) $\mathrm{Data\text{-}Loc}_{\mathrm{SCM}} \subseteq \mathrm{dom}\, s$.

(22) $\mathrm{dom}(s{\restriction}\mathrm{Data\text{-}Loc}_{\mathrm{SCM}}) = \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$.

(23) If $p = q$, then $\mathrm{DataPart}(p) = \mathrm{DataPart}(q)$.

(24) $\mathrm{DataPart}(p) = p\restriction \text{Data-Loc}_{\mathrm{SCM}}$.

(25) p is data-only iff $\mathrm{dom}\, p \subseteq \text{Data-Loc}_{\mathrm{SCM}}$.

(26) $\mathrm{dom}\,\mathrm{DataPart}(p) \subseteq \text{Data-Loc}_{\mathrm{SCM}}$.

(27) $\text{Instr-Loc}_{\mathrm{SCM}} \subseteq \mathrm{dom}\, s$.

(28) If $p = q$, then $\mathrm{ProgramPart}(p) = \mathrm{ProgramPart}(q)$.

(29) $\mathrm{dom}\,\mathrm{ProgramPart}(p) \subseteq \text{Instr-Loc}_{\mathrm{SCM}}$.

Let us consider R and let I be an element of the instructions of $\mathbf{SCM}(R)$. Observe that $\mathrm{InsCode}(I)$ is natural.

Next we state several propositions:

(30) $\mathrm{InsCode}(I) \leqslant 7$.

(31) $\mathrm{IncAddr}(\text{goto } l_1, k) = \text{goto } (l_1 + k)$.

(32) $\mathrm{IncAddr}(\textbf{if } a = 0 \textbf{ goto } l_1, k) = \textbf{if } a = 0 \textbf{ goto } l_1 + k$.

(33) $s(a) = (s+\cdot \text{Start-At}(l_1))(a)$.

(34) Suppose $\mathbf{IC}_{(s_1)} = \mathbf{IC}_{(s_2)}$ and for every Data-Location a of R holds $s_1(a) = s_2(a)$ and for every instruction-location i of $\mathbf{SCM}(R)$ holds $s_1(i) = s_2(i)$. Then $s_1 = s_2$.

(35) $\mathrm{Exec}(\mathrm{IncAddr}(\mathrm{CurInstr}(s), k), s+\cdot \text{Start-At}(\mathbf{IC}_s + k)) = \mathrm{Following}(s)+\cdot \text{Start-At}(\mathbf{IC}_{\mathrm{Following}(s)} + k)$.

(36) If $\mathbf{IC}_s = \mathrm{il}_{\mathbf{SCM}(R)}(j + k)$, then $\mathrm{Exec}(I, s+\cdot \text{Start-At}(\mathbf{IC}_s -' k)) = \mathrm{Exec}(\mathrm{IncAddr}(I, k), s)+\cdot \text{Start-At}(\mathbf{IC}_{\mathrm{Exec}(\mathrm{IncAddr}(I,k),s)} -' k)$.

Let us consider R. One can check that there exists a finite partial state of $\mathbf{SCM}(R)$ which is autonomic and non programmed.

Let us consider R, let a be a Data-Location of R, and let r be an element of the carrier of R. Then $a \mathrel{\dot{\longmapsto}} r$ is a finite partial state of $\mathbf{SCM}(R)$.

We now state a number of propositions:

(37) If R is non trivial, then for every autonomic finite partial state p of $\mathbf{SCM}(R)$ such that $\mathrm{DataPart}(p) \neq \emptyset$ holds $\mathbf{IC}_{\mathbf{SCM}(R)} \in \mathrm{dom}\, p$.

(38) If R is non trivial, then for every autonomic non programmed finite partial state p of $\mathbf{SCM}(R)$ holds $\mathbf{IC}_{\mathbf{SCM}(R)} \in \mathrm{dom}\, p$.

(39) For every autonomic finite partial state p of $\mathbf{SCM}(R)$ such that $\mathbf{IC}_{\mathbf{SCM}(R)} \in \mathrm{dom}\, p$ holds $\mathbf{IC}_p \in \mathrm{dom}\, p$.

(40) Suppose R is non trivial. Let p be an autonomic non programmed finite partial state of $\mathbf{SCM}(R)$. If $p \subseteq s$, then $\mathbf{IC}_{(\mathrm{Computation}(s))(n)} \in \mathrm{dom}\,\mathrm{ProgramPart}(p)$.

(41) Suppose R is non trivial. Let p be an autonomic non programmed finite partial state of $\mathbf{SCM}(R)$. If $p \subseteq s_1$ and $p \subseteq s_2$, then $\mathbf{IC}_{(\mathrm{Computation}(s_1))(n)} = \mathbf{IC}_{(\mathrm{Computation}(s_2))(n)}$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(n)) = \mathrm{CurInstr}((\mathrm{Computation}(s_2))(n))$.

(42) Suppose R is non trivial. Let p be an autonomic non programmed finite partial state of $\mathbf{SCM}(R)$. If $p \subseteq s_1$ and $p \subseteq s_2$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(n)) = a{:=}b$ and $a \in \mathrm{dom}\, p$, then $(\mathrm{Computation}(s_1))(n)(b) = (\mathrm{Computation}(s_2))(n)(b)$.

(43) Suppose R is non trivial. Let p be an autonomic non programmed finite partial state of $\mathbf{SCM}(R)$. Suppose $p \subseteq s_1$ and $p \subseteq s_2$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(n)) = \mathrm{AddTo}(a, b)$ and $a \in \mathrm{dom}\, p$. Then $(\mathrm{Computation}(s_1))(n)(a) + (\mathrm{Computation}(s_1))(n)(b) = (\mathrm{Computation}(s_2))(n)(a) + (\mathrm{Computation}(s_2))(n)(b)$.

(44) Suppose R is non trivial. Let p be an autonomic non programmed finite partial state of $\mathbf{SCM}(R)$. Suppose $p \subseteq s_1$ and $p \subseteq s_2$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(n)) = \mathrm{SubFrom}(a, b)$ and $a \in \mathrm{dom}\, p$. Then $(\mathrm{Computation}(s_1))(n)(a) - (\mathrm{Computation}(s_1))(n)(b) = (\mathrm{Computation}(s_2))(n)(a) - (\mathrm{Computation}(s_2))(n)(b)$.

(45) Suppose R is non trivial. Let p be an autonomic non programmed finite partial state of $\mathbf{SCM}(R)$. Suppose $p \subseteq s_1$ and $p \subseteq s_2$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(n)) = \mathrm{MultBy}(a, b)$ and $a \in \mathrm{dom}\, p$. Then $(\mathrm{Computation}(s_1))(n)(a) \cdot (\mathrm{Computation}(s_1))(n)(b) = (\mathrm{Computation}(s_2))(n)(a) \cdot (\mathrm{Computation}(s_2))(n)(b)$.

(46) Suppose R is non trivial. Let p be an autonomic non programmed finite partial state of $\mathbf{SCM}(R)$. Suppose $p \subseteq s_1$ and $p \subseteq s_2$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(n)) = \mathbf{if}\ a = 0\ \mathbf{goto}\ l_1$ and $l_1 \neq \mathrm{Next}(\mathbf{IC}_{(\mathrm{Computation}(s_1))(n)})$. Then $(\mathrm{Computation}(s_1))(n)(a) = 0_R$ if and only if $(\mathrm{Computation}(s_2))(n)(a) = 0_R$.

3. Relocability

Let N be a set with non empty elements, let S be a regular standard IC-Ins-separated definite non empty non void AMI over N, let k be a natural number, and let p be a finite partial state of S. The functor $\mathrm{Relocated}(p, k)$ yielding a finite partial state of S is defined as follows:

(Def. 2) $\mathrm{Relocated}(p, k) = \mathrm{Start\text{-}At}(\mathbf{IC}_p + k) +\cdot \mathrm{IncAddr}(\mathrm{Shift}(\mathrm{ProgramPart}(p), k), k) +\cdot \mathrm{DataPart}(p)$.

In the sequel S denotes a regular standard IC-Ins-separated definite non empty non void AMI over N, g denotes a finite partial state of S, and i_1 denotes an instruction-location of S.

One can prove the following propositions:

(47) $\mathrm{DataPart}(\mathrm{Relocated}(g, k)) = \mathrm{DataPart}(g)$.

(48) If S is realistic, then $\mathrm{ProgramPart}(\mathrm{Relocated}(g, k)) = \mathrm{IncAddr}(\mathrm{Shift}(\mathrm{ProgramPart}(g), k), k)$.

(49) If S is realistic, then $\operatorname{dom}\operatorname{ProgramPart}(\operatorname{Relocated}(g,k)) = \{\operatorname{il}_S(j+k); j$ ranges over natural numbers: $\operatorname{il}_S(j) \in \operatorname{dom}\operatorname{ProgramPart}(g)\}$.

(50) If S is realistic, then $i_1 \in \operatorname{dom} g$ iff $i_1 + k \in \operatorname{dom}\operatorname{Relocated}(g,k)$.

(51) $\mathbf{IC}_S \in \operatorname{dom}\operatorname{Relocated}(g,k)$.

(52) If S is realistic, then $\mathbf{IC}_{\operatorname{Relocated}(g,k)} = \mathbf{IC}_g + k$.

(53) Let p be a programmed finite partial state of S and l be an instruction-location of S. If $l \in \operatorname{dom} p$, then $(\operatorname{IncAddr}(p,k))(l) = \operatorname{IncAddr}(\pi_l p, k)$.

(54) For every programmed finite partial state p of S holds $\operatorname{Shift}(\operatorname{IncAddr}(p,i),i) = \operatorname{IncAddr}(\operatorname{Shift}(p,i),i)$.

(55) If S is realistic, then for every instruction I of S such that $i_1 \in \operatorname{dom}\operatorname{ProgramPart}(g)$ and $I = g(i_1)$ holds $\operatorname{IncAddr}(I,k) = (\operatorname{Relocated}(g,k))(i_1+k)$.

(56) If S is realistic, then $\text{Start-At}(\mathbf{IC}_g + k) \subseteq \operatorname{Relocated}(g,k)$.

(57) If S is realistic, then for every data-only finite partial state q of S such that $\mathbf{IC}_S \in \operatorname{dom} g$ holds $\operatorname{Relocated}(g{+\cdot}q,k) = \operatorname{Relocated}(g,k){+\cdot}q$.

(58) For every autonomic finite partial state p of $\mathbf{SCM}(R)$ such that $p \subseteq s_1$ and $\operatorname{Relocated}(p,k) \subseteq s_2$ holds $p \subseteq s_1{+\cdot}s_2{\restriction}\text{Data-Loc}_{\mathrm{SCM}}$.

(59) Suppose R is non trivial. Let p be an autonomic finite partial state of $\mathbf{SCM}(R)$. Suppose $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$ and $p \subseteq s_1$ and $\operatorname{Relocated}(p,k) \subseteq s_2$ and $s = s_1{+\cdot}s_2{\restriction}\text{Data-Loc}_{\mathrm{SCM}}$. Let i be a natural number. Then $\mathbf{IC}_{(\operatorname{Computation}(s_1))(i)} + k = \mathbf{IC}_{(\operatorname{Computation}(s_2))(i)}$ and $\operatorname{IncAddr}(\operatorname{CurInstr}((\operatorname{Computation}(s_1))(i)),k) = \operatorname{CurInstr}((\operatorname{Computation}(s_2))(i))$ and $(\operatorname{Computation}(s_1))(i){\restriction}\operatorname{dom}\operatorname{DataPart}(p) = (\operatorname{Computation}(s_2))(i){\restriction}\operatorname{dom}\operatorname{DataPart}(\operatorname{Relocated}(p,k))$ and $(\operatorname{Computation}(s))(i){\restriction}\text{Data-Loc}_{\mathrm{SCM}} = (\operatorname{Computation}(s_2))(i){\restriction}\text{Data-Loc}_{\mathrm{SCM}}$.

(60) Suppose R is non trivial. Let p be an autonomic finite partial state of $\mathbf{SCM}(R)$. If $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$, then p is halting iff $\operatorname{Relocated}(p,k)$ is halting.

(61) Suppose R is non trivial. Let p be an autonomic finite partial state of $\mathbf{SCM}(R)$. Suppose $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$ and $p \subseteq s$. Let i be a natural number. Then $(\operatorname{Computation}(s{+\cdot}\operatorname{Relocated}(p,k)))(i) = (\operatorname{Computation}(s))(i){+\cdot}\operatorname{Start-At}(\mathbf{IC}_{(\operatorname{Computation}(s))(i)} + k){+\cdot}\operatorname{ProgramPart}(\operatorname{Relocated}(p,k))$.

(62) Suppose R is non trivial. Let p be an autonomic finite partial state of $\mathbf{SCM}(R)$. Suppose $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$ and $\operatorname{Relocated}(p,k) \subseteq s$. Let i be a natural number. Then $(\operatorname{Computation}(s))(i) = (\operatorname{Computation}(s{+\cdot}p))(i){+\cdot}\operatorname{Start-At}(\mathbf{IC}_{(\operatorname{Computation}(s{+\cdot}p))(i)} + k){+\cdot}s{\restriction}\operatorname{dom}\operatorname{ProgramPart}(p){+\cdot}\operatorname{ProgramPart}(\operatorname{Relocated}(p,k))$.

(63) Suppose R is non trivial and $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$ and $p \subseteq s$ and $\operatorname{Relocated}(p,k)$ is autonomic. Let i be a natural number. Then

(Computation(s))(i) = (Computation($s+\cdot$ Relocated(p, k)))(i)$+\cdot$ Start-At ($\mathbf{IC}_{(\text{Computation}(s+\cdot \text{Relocated}(p,k)))(i)} -' k)+\cdot s\restriction \operatorname{dom}$ ProgramPart(Relocated(p, k))$+\cdot$ ProgramPart(p).

(64) If R is non trivial and $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$, then p is autonomic iff Relocated(p, k) is autonomic.

(65) Suppose R is non trivial. Let p be a halting autonomic finite partial state of $\mathbf{SCM}(R)$. If $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$, then DataPart(Result($p$)) = DataPart(Result(Relocated(p, k))).

(66) Suppose R is non trivial. Let F be a partial function from FinPartSt($\mathbf{SCM}(R)$) to FinPartSt($\mathbf{SCM}(R)$). Suppose $\mathbf{IC}_{\mathbf{SCM}(R)} \in \operatorname{dom} p$ and F is data-only. Then p computes F if and only if Relocated(p, k) computes F.

References

[1] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(**1**):41–46, 1990.
[2] Grzegorz Bancerek. König's theorem. *Formalized Mathematics*, 1(**3**):589–593, 1990.
[3] Grzegorz Bancerek. The ordinal numbers. *Formalized Mathematics*, 1(**1**):91–96, 1990.
[4] Grzegorz Bancerek. Sequences of ordinal numbers. *Formalized Mathematics*, 1(**2**):281–290, 1990.
[5] Józef Białas. Group and field definitions. *Formalized Mathematics*, 1(**3**):433–439, 1990.
[6] Czesław Byliński. A classical first order language. *Formalized Mathematics*, 1(**4**):669–676, 1990.
[7] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.
[8] Czesław Byliński. Functions from a set to a set. *Formalized Mathematics*, 1(**1**):153–164, 1990.
[9] 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.
[10] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(**1**):165–167, 1990.
[11] Artur Korniłowicz. The basic properties of **SCM** over ring. *Formalized Mathematics*, 7(**2**):301–305, 1998.
[12] Artur Korniłowicz. The construction of **SCM** over ring. *Formalized Mathematics*, 7(**2**):295–300, 1998.
[13] Artur Korniłowicz. On the composition of macro instructions of standard computers. *Formalized Mathematics*, 9(**2**):303–316, 2001.
[14] Artur Korniłowicz. The properties of instructions of SCM over ring. *Formalized Mathematics*, 9(**2**):317–322, 2001.
[15] Eugeniusz Kusak, Wojciech Leończuk, and Michał Muzalewski. Abelian groups, fields and vector spaces. *Formalized Mathematics*, 1(**2**):335–342, 1990.
[16] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Formalized Mathematics*, 3(**2**):151–160, 1992.
[17] Yatsuka Nakamura and Andrzej Trybulec. On a mathematical model of programs. *Formalized Mathematics*, 3(**2**):241–250, 1992.
[18] Takaya Nishiyama and Yasuho Mizuhara. Binary arithmetics. *Formalized Mathematics*, 4(**1**):83–86, 1993.
[19] Dariusz Surowik. Cyclic groups and some of their properties - part I. *Formalized Mathematics*, 2(**5**):623–627, 1991.
[20] Yasushi Tanaka. On the decomposition of the states of SCM. *Formalized Mathematics*, 5(**1**):1–8, 1996.
[21] Andrzej Trybulec. Subsets of complex numbers. *To appear in Formalized Mathematics*.

[22] Andrzej Trybulec. Binary operations applied to functions. *Formalized Mathematics*, 1(**2**):329–334, 1990.
[23] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[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, Piotr Rudnicki, and Artur Korniłowicz. Standard ordering of instruction locations. *Formalized Mathematics*, 9(**2**):291–301, 2001.
[26] Wojciech A. Trybulec. Vectors in real linear space. *Formalized Mathematics*, 1(**2**):291–296, 1990.
[27] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.
[28] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.

Received February 6, 2004