FORMALIZED MATHEMATICS
Volume 9, Number 2, 2001
University of Białystok

The Properties of Instructions of SCM over Ring

Artur Korniłowicz
University of Białystok

MML Identifier: SCMRING3.

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

For simplicity, we adopt the following convention: R denotes a good ring, r denotes an element of the carrier of R, a, b denote Data-Locations of R, i_1, i_2, i_3 denote instruction-locations of $\mathbf{SCM}(R)$, I denotes an instruction of $\mathbf{SCM}(R)$, s_1, s_2 denote states of $\mathbf{SCM}(R)$, T denotes an instruction type of $\mathbf{SCM}(R)$, and k denotes a natural number.

Let us note that $\mathbb{Z}$ is infinite.

One can verify that INT.Ring is infinite and good.

Let us mention that there exists a 1-sorted structure which is strict and infinite.

Let us mention that there exists a ring which is strict, infinite, and good.

We now state the proposition

(1) $\mathrm{ObjectKind}(a) =$ the carrier of R.

Let R be a good ring, let l_1, l_2 be Data-Locations of R, and let a, b be elements of R. Then $[l_1 \longmapsto a, l_2 \longmapsto b]$ is a finite partial state of $\mathbf{SCM}(R)$.

We now state a number of propositions:

(2) $a \notin$ the instruction locations of $\mathbf{SCM}(R)$.

(3) $a \neq \mathbf{IC}_{\mathbf{SCM}(R)}$.

(4) $\mathrm{Data\text{-}Loc}_{\mathrm{SCM}} \neq$ the instruction locations of $\mathbf{SCM}(R)$.

(5) For every object o of $\mathbf{SCM}(R)$ holds $o = \mathbf{IC}_{\mathbf{SCM}(R)}$ or $o \in$ the instruction locations of $\mathbf{SCM}(R)$ or o is a Data-Location of R.

(6) If $i_2 \neq i_3$, then $\mathrm{Next}(i_2) \neq \mathrm{Next}(i_3)$.

ISSN 1426–2630

(7) If s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}(R)$, then $s_1(a) = s_2(a)$.

(8) $\mathrm{InsCode}(\mathbf{halt}_{\mathbf{SCM}(R)}) = 0$.

(9) $\mathrm{InsCode}(a{:=}b) = 1$.

(10) $\mathrm{InsCode}(\mathrm{AddTo}(a, b)) = 2$.

(11) $\mathrm{InsCode}(\mathrm{SubFrom}(a, b)) = 3$.

(12) $\mathrm{InsCode}(\mathrm{MultBy}(a, b)) = 4$.

(13) $\mathrm{InsCode}(a{:=}r) = 5$.

(14) $\mathrm{InsCode}(\mathrm{goto}\ i_2) = 6$.

(15) $\mathrm{InsCode}(\mathbf{if}\ a = 0\ \mathbf{goto}\ i_2) = 7$.

(16) If $\mathrm{InsCode}(I) = 0$, then $I = \mathbf{halt}_{\mathbf{SCM}(R)}$.

(17) If $\mathrm{InsCode}(I) = 1$, then there exist a, b such that $I = a{:=}b$.

(18) If $\mathrm{InsCode}(I) = 2$, then there exist a, b such that $I = \mathrm{AddTo}(a, b)$.

(19) If $\mathrm{InsCode}(I) = 3$, then there exist a, b such that $I = \mathrm{SubFrom}(a, b)$.

(20) If $\mathrm{InsCode}(I) = 4$, then there exist a, b such that $I = \mathrm{MultBy}(a, b)$.

(21) If $\mathrm{InsCode}(I) = 5$, then there exist a, r such that $I = a{:=}r$.

(22) If $\mathrm{InsCode}(I) = 6$, then there exists i_3 such that $I = \mathrm{goto}\ i_3$.

(23) If $\mathrm{InsCode}(I) = 7$, then there exist a, i_2 such that $I = \mathbf{if}\ a = 0\ \mathbf{goto}\ i_2$.

(24) $\mathrm{AddressPart}(\mathbf{halt}_{\mathbf{SCM}(R)}) = \varepsilon$.

(25) $\mathrm{AddressPart}(a{:=}b) = \langle a, b\rangle$.

(26) $\mathrm{AddressPart}(\mathrm{AddTo}(a, b)) = \langle a, b\rangle$.

(27) $\mathrm{AddressPart}(\mathrm{SubFrom}(a, b)) = \langle a, b\rangle$.

(28) $\mathrm{AddressPart}(\mathrm{MultBy}(a, b)) = \langle a, b\rangle$.

(29) $\mathrm{AddressPart}(a{:=}r) = \langle a, r\rangle$.

(30) $\mathrm{AddressPart}(\mathrm{goto}\ i_2) = \langle i_2\rangle$.

(31) $\mathrm{AddressPart}(\mathbf{if}\ a = 0\ \mathbf{goto}\ i_2) = \langle i_2, a\rangle$.

(32) If $T = 0$, then $\mathrm{AddressParts}\, T = \{0\}$.

Let us consider R, T. Observe that $\mathrm{AddressParts}\, T$ is non empty.

We now state a number of propositions:

(33) If $T = 1$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1, 2\}$.

(34) If $T = 2$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1, 2\}$.

(35) If $T = 3$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1, 2\}$.

(36) If $T = 4$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1, 2\}$.

(37) If $T = 5$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1, 2\}$.

(38) If $T = 6$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1\}$.

(39) If $T = 7$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1, 2\}$.

(40) $\prod_{\mathrm{AddressParts}\, \mathrm{InsCode}(a{:=}b)}(1) = \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$.

(41) $\prod_{\mathrm{AddressParts\,InsCode}(a:=b)}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(42) $\prod_{\mathrm{AddressParts\,InsCode}(\mathrm{AddTo}(a,b))}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(43) $\prod_{\mathrm{AddressParts\,InsCode}(\mathrm{AddTo}(a,b))}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(44) $\prod_{\mathrm{AddressParts\,InsCode}(\mathrm{SubFrom}(a,b))}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(45) $\prod_{\mathrm{AddressParts\,InsCode}(\mathrm{SubFrom}(a,b))}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(46) $\prod_{\mathrm{AddressParts\,InsCode}(\mathrm{MultBy}(a,b))}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(47) $\prod_{\mathrm{AddressParts\,InsCode}(\mathrm{MultBy}(a,b))}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(48) $\prod_{\mathrm{AddressParts\,InsCode}(a:=r)}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(49) $\prod_{\mathrm{AddressParts\,InsCode}(a:=r)}(2) =$ the carrier of R.

(50) $\prod_{\mathrm{AddressParts\,InsCode}(\mathrm{goto}\ i_2)}(1) =$ the instruction locations of $\mathbf{SCM}(R)$.

(51) $\prod_{\mathrm{AddressParts\,InsCode}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2)}(1) =$ the instruction locations of $\mathbf{SCM}(R)$.

(52) $\prod_{\mathrm{AddressParts\,InsCode}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2)}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(53) $\mathrm{NIC}(\mathbf{halt}_{\mathbf{SCM}(R)}, i_1) = \{i_1\}$.

Let us consider R. One can check that $\mathrm{JUMP}(\mathbf{halt}_{\mathbf{SCM}(R)})$ is empty.

Next we state the proposition

(54) $\mathrm{NIC}(a:=b, i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider R, a, b. Observe that $\mathrm{JUMP}(a:=b)$ is empty.

We now state the proposition

(55) $\mathrm{NIC}(\mathrm{AddTo}(a,b), i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider R, a, b. One can check that $\mathrm{JUMP}(\mathrm{AddTo}(a,b))$ is empty.

One can prove the following proposition

(56) $\mathrm{NIC}(\mathrm{SubFrom}(a,b), i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider R, a, b. Note that $\mathrm{JUMP}(\mathrm{SubFrom}(a,b))$ is empty.

Next we state the proposition

(57) $\mathrm{NIC}(\mathrm{MultBy}(a,b), i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider R, a, b. One can verify that $\mathrm{JUMP}(\mathrm{MultBy}(a,b))$ is empty.

One can prove the following proposition

(58) $\mathrm{NIC}(a:=r, i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider R, a, r. Note that $\mathrm{JUMP}(a:=r)$ is empty.

The following propositions are true:

(59) $\mathrm{NIC}(\mathrm{goto}\ i_2, i_1) = \{i_2\}$.

(60) $\mathrm{JUMP}(\mathrm{goto}\ i_2) = \{i_2\}$.

Let us consider R, i_2. Note that $\mathrm{JUMP}(\mathrm{goto}\ i_2)$ is non empty and trivial.

We now state two propositions:

(61) $i_2 \in \mathrm{NIC}(\mathbf{if}\ a = 0\ \mathbf{goto}\ i_2, i_1)$ and $\mathrm{NIC}(\mathbf{if}\ a = 0\ \mathbf{goto}\ i_2, i_1) \subseteq \{i_2, \mathrm{Next}(i_1)\}$.

(62) $\mathrm{JUMP}(\mathbf{if}\ a = 0\ \mathbf{goto}\ i_2) = \{i_2\}$.

Let us consider R, a, i_2. Observe that JUMP(**if** $a = 0$ **goto** i_2) is non empty and trivial.

One can prove the following two propositions:

(63) $\mathrm{SUCC}(i_1) = \{i_1, \mathrm{Next}(i_1)\}$.

(64) Let f be a function from $\mathbb{N}$ into the instruction locations of $\mathbf{SCM}(R)$. Suppose that for every natural number k holds $f(k) = \mathbf{i}_k$. Then

(i) f is bijective, and

(ii) for every natural number k holds $f(k+1) \in \mathrm{SUCC}(f(k))$ and for every natural number j such that $f(j) \in \mathrm{SUCC}(f(k))$ holds $k \leqslant j$.

Let us consider R. Note that $\mathbf{SCM}(R)$ is standard.

Next we state three propositions:

(65) $\mathrm{il}_{\mathbf{SCM}(R)}(k) = \mathbf{i}_k$.

(66) $\mathrm{Next}(\mathrm{il}_{\mathbf{SCM}(R)}(k)) = \mathrm{il}_{\mathbf{SCM}(R)}(k+1)$.

(67) $\mathrm{Next}(i_1) = \mathrm{NextLoc}\, i_1$.

Let R be a good ring and let k be a natural number. The functor $\mathrm{dl}_R(k)$ yields a Data-Location of R and is defined as follows:

(Def. 1) $\mathrm{dl}_R(k) = \mathbf{d}_k$.

Let us consider R. Observe that InsCode($\mathbf{halt}_{\mathbf{SCM}(R)}$) is jump-only.

Let us consider R. Note that $\mathbf{halt}_{\mathbf{SCM}(R)}$ is jump-only.

Let us consider R, i_2. Note that InsCode(goto i_2) is jump-only.

Let us consider R, i_2. One can check that goto i_2 is jump-only.

Let us consider R, a, i_2. Observe that InsCode(**if** $a = 0$ **goto** i_2) is jump-only.

Let us consider R, a, i_2. Note that **if** $a = 0$ **goto** i_2 is jump-only.

In the sequel S denotes a non trivial good ring, p, q denote Data-Locations of S, and w denotes an element of the carrier of S.

Let us consider S, p, q. One can check that InsCode(p:=q) is non jump-only.

Let us consider S, p, q. One can check that p:=q is non jump-only.

Let us consider S, p, q. Observe that InsCode(AddTo(p, q)) is non jump-only.

Let us consider S, p, q. Note that AddTo(p, q) is non jump-only.

Let us consider S, p, q. Note that InsCode(SubFrom(p, q)) is non jump-only.

Let us consider S, p, q. Note that SubFrom(p, q) is non jump-only.

Let us consider S, p, q. Observe that InsCode(MultBy(p, q)) is non jump-only.

Let us consider S, p, q. One can verify that MultBy(p, q) is non jump-only.

Let us consider S, p, w. Note that InsCode(p:=w) is non jump-only.

Let us consider S, p, w. Note that p:=w is non jump-only.

Let us consider R, a, b. Observe that a:=b is sequential.

Let us consider R, a, b. Observe that AddTo(a, b) is sequential.

Let us consider R, a, b. Note that SubFrom(a, b) is sequential.

Let us consider R, a, b. One can verify that MultBy(a, b) is sequential.

Let us consider R, a, r. Note that $a{:=}r$ is sequential.

Let us consider R, i_2. One can check that goto i_2 is non sequential.

Let us consider R, a, i_2. Observe that **if** $a = 0$ **goto** i_2 is non sequential.

Let us consider R, i_2. Note that goto i_2 is non instruction location free.

Let us consider R, a, i_2. Note that **if** $a = 0$ **goto** i_2 is non instruction location free.

Let us consider R. One can check that $\mathbf{SCM}(R)$ is homogeneous and explicit-jump-instruction and has ins-loc-in-jump.

Let us consider R. Observe that $\mathbf{SCM}(R)$ is regular.

Next we state two propositions:

(68) $\mathrm{IncAddr}(\mathrm{goto}\ i_2, k) = \mathrm{goto}\ \mathrm{il}_{\mathbf{SCM}(R)}(\mathrm{locnum}(i_2) + k)$.

(69) $\mathrm{IncAddr}(\mathbf{if}\ a = 0\ \mathbf{goto}\ i_2, k) = \mathbf{if}\ a = 0\ \mathbf{goto}\ \mathrm{il}_{\mathbf{SCM}(R)}(\mathrm{locnum}(i_2) + k)$.

Let us consider R. One can check that $\mathbf{SCM}(R)$ is IC-good and Exec-preserving.

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 and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 1990.

[4] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Formalized Mathematics*, 5(**4**):485–492, 1996.

[5] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.

[6] Czesław Byliński. Functions from a set to a set. *Formalized Mathematics*, 1(**1**):153–164, 1990.

[7] Artur Korniłowicz. The basic properties of **SCM** over ring. *Formalized Mathematics*, 7(**2**):301–305, 1998.

[8] Artur Korniłowicz. The construction of **SCM** over ring. *Formalized Mathematics*, 7(**2**):295–300, 1998.

[9] Artur Korniłowicz. On the composition of macro instructions of standard computers. *Formalized Mathematics*, 9(**2**):303–316, 2001.

[10] Eugeniusz Kusak, Wojciech Leończuk, and Michał Muzalewski. Abelian groups, fields and vector spaces. *Formalized Mathematics*, 1(**2**):335–342, 1990.

[11] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Formalized Mathematics*, 3(**2**):151–160, 1992.

[12] Yatsuka Nakamura and Andrzej Trybulec. On a mathematical model of programs. *Formalized Mathematics*, 3(**2**):241–250, 1992.

[13] Yozo Toda. The formalization of simple graphs. *Formalized Mathematics*, 5(**1**):137–144, 1996.

[14] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.

[15] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Formalized Mathematics*, 4(**1**):51–56, 1993.

[16] Andrzej Trybulec, Piotr Rudnicki, and Artur Korniłowicz. Standard ordering of instruction locations. *Formalized Mathematics*, 9(**2**):291–301, 2001.

[17] Wojciech A. Trybulec. Vectors in real linear space. *Formalized Mathematics*, 1(**2**):291–296, 1990.

[18] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.

[19] Zinaida Trybulec and Halina Święczkowska. Boolean properties of sets. *Formalized Mathematics*, 1(**1**):17–23, 1990.
[20] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.
[21] Edmund Woronowicz. Relations defined on sets. *Formalized Mathematics*, 1(**1**):181–186, 1990.

Received April 14, 2000