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

Modifying Addresses of Instructions of $\mathbf{SCM}_{\rm FSA}$

Andrzej Trybulec
Warsaw University
Białystok

Yatsuka Nakamura
Shinshu University
Nagano

MML Identifier: SCMFSA_4.

WWW: http://mizar.org/JFM/Vol8/scmfsa_4.html

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

1. Preliminaries

Let N be a set and let S be an AMI over N. Observe that every finite partial state of S is finite.

Let N be a set and let S be an AMI over N. One can verify that there exists a finite partial state of S which is programmed.

We now state the proposition

(1) Let N be a set with non empty elements, S be a definite non empty non void AMI over N, and p be a programmed finite partial state of S. Then $\operatorname{rng} p \subseteq$ the instructions of S.

Let N be a set, let S be an AMI over N, and let I, J be programmed finite partial states of S. Then $I+\cdot J$ is a programmed finite partial state of S.

We now state the proposition

(2) Let N be a set with non empty elements, S be a definite non empty non void AMI over N, f be a function from the instructions of S into the instructions of S, and s be a programmed finite partial state of S. Then $\operatorname{dom}(f \cdot s) = \operatorname{dom} s$.

2. Incrementing and decrementing the instruction locations

In the sequel k, l, m, n, p denote natural numbers.

Let l_1 be an instruction-location of $\mathbf{SCM}_{\rm FSA}$ and let k be a natural number. The functor $l_1 + k$ yields an instruction-location of $\mathbf{SCM}_{\rm FSA}$ and is defined as follows:

(Def. 1) There exists a natural number m such that $l_1 = \operatorname{insloc}(m)$ and $l_1 + k = \operatorname{insloc}(m+k)$.

The functor $l_1 -' k$ yields an instruction-location of $\mathbf{SCM}_{\rm FSA}$ and is defined as follows:

(Def. 2) There exists a natural number m such that $l_1 = \operatorname{insloc}(m)$ and $l_1 -' k = \operatorname{insloc}(m -' k)$.

One can prove the following propositions:

(3) For every instruction-location l of $\mathbf{SCM}_{\rm FSA}$ and for all m, n holds $(l+m)+n = l+(m+n)$.

(4) For every instruction-location l_1 of $\mathbf{SCM}_{\rm FSA}$ and for every natural number k holds $(l_1 + k) -' k = l_1$.

In the sequel L is an instruction-location of **SCM** and I is an instruction of **SCM**.
We now state three propositions:

(5) For every instruction-location l of $\mathbf{SCM}_{\rm FSA}$ and for every L such that $L = l$ holds $l + k = L + k$.

(6) For all instruction-locations l_2, l_3 of $\mathbf{SCM}_{\rm FSA}$ and for every natural number k holds $\text{Start-At}(l_2 + k) = \text{Start-At}(l_3 + k)$ iff $\text{Start-At}(l_2) = \text{Start-At}(l_3)$.

(7) For all instruction-locations l_2, l_3 of $\mathbf{SCM}_{\rm FSA}$ and for every natural number k such that $\text{Start-At}(l_2) = \text{Start-At}(l_3)$ holds $\text{Start-At}(l_2 -' k) = \text{Start-At}(l_3 -' k)$.

3. Incrementing addresses

Let i be an instruction of $\mathbf{SCM}_{\rm FSA}$ and let k be a natural number. The functor $\text{IncAddr}(i,k)$ yielding an instruction of $\mathbf{SCM}_{\rm FSA}$ is defined as follows:

(Def. 3)(i) There exists an instruction I of **SCM** such that $I = i$ and $\text{IncAddr}(i,k) = \text{IncAddr}(I,k)$ if $\text{InsCode}(i) \in \{6,7,8\}$,

(ii) $\text{IncAddr}(i,k) = i$, otherwise.

The following propositions are true:

(8) For every natural number k holds $\text{IncAddr}(\mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}, k) = \mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$.

(9) For every natural number k and for all integer locations a, b holds $\text{IncAddr}(a{:=}b, k) = a{:=}b$.

(10) For every natural number k and for all integer locations a, b holds $\text{IncAddr}(\text{AddTo}(a,b),k) = \text{AddTo}(a,b)$.

(11) For every natural number k and for all integer locations a, b holds $\text{IncAddr}(\text{SubFrom}(a,b),k) = \text{SubFrom}(a,b)$.

(12) For every natural number k and for all integer locations a, b holds $\text{IncAddr}(\text{MultBy}(a,b),k) = \text{MultBy}(a,b)$.

(13) For every natural number k and for all integer locations a, b holds $\text{IncAddr}(\text{Divide}(a,b),k) = \text{Divide}(a,b)$.

(14) For every natural number k and for every instruction-location l_1 of $\mathbf{SCM}_{\rm FSA}$ holds $\text{IncAddr}(\text{goto } l_1, k) = \text{goto } (l_1 + k)$.

(15) Let k be a natural number, l_1 be an instruction-location of $\mathbf{SCM}_{\rm FSA}$, and a be an integer location. Then $\text{IncAddr}(\mathbf{if}\ a = 0\ \mathbf{goto}\ l_1, k) = \mathbf{if}\ a = 0\ \mathbf{goto}\ l_1 + k$.

(16) Let k be a natural number, l_1 be an instruction-location of $\mathbf{SCM}_{\rm FSA}$, and a be an integer location. Then $\text{IncAddr}(\mathbf{if}\ a > 0\ \mathbf{goto}\ l_1, k) = \mathbf{if}\ a > 0\ \mathbf{goto}\ l_1 + k$.

(17) Let k be a natural number, a, b be integer locations, and f be a finite sequence location. Then $\text{IncAddr}(b{:=}f_a, k) = b{:=}f_a$.

(18) Let k be a natural number, a, b be integer locations, and f be a finite sequence location. Then $\text{IncAddr}(f_a{:=}b, k) = f_a{:=}b$.

(19) Let k be a natural number, a be an integer location, and f be a finite sequence location. Then $\text{IncAddr}(a{:=}\text{len} f, k) = a{:=}\text{len} f$.

(20) Let k be a natural number, a be an integer location, and f be a finite sequence location. Then $\text{IncAddr}(f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle, k) = f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle$.

(21) For every instruction i of $\mathbf{SCM}_{\rm FSA}$ and for every I such that $i = I$ holds $\mathrm{IncAddr}(i,k) = \mathrm{IncAddr}(I,k)$.

(22) For every instruction I of $\mathbf{SCM}_{\rm FSA}$ and for every natural number k holds $\mathrm{InsCode}(\mathrm{IncAddr}(I,k)) = \mathrm{InsCode}(I)$.

Let I_1 be a finite partial state of $\mathbf{SCM}_{\rm FSA}$. We say that I_1 is initial if and only if:

(Def. 4) For all m, n such that $\mathrm{insloc}(n) \in \mathrm{dom}\, I_1$ and $m < n$ holds $\mathrm{insloc}(m) \in \mathrm{dom}\, I_1$.

The finite partial state $\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}}$ of $\mathbf{SCM}_{\rm FSA}$ is defined by:

(Def. 5) $\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}} = \mathrm{insloc}(0) \dot{\longmapsto} \mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$.

Let us note that $\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}}$ is non empty, initial, and programmed.

Let us observe that there exists a finite partial state of $\mathbf{SCM}_{\rm FSA}$ which is initial, programmed, and non empty.

Let f be a function and let g be a finite function. Note that $f \cdot g$ is finite.

Let N be a non empty set with non empty elements, let S be a definite non empty non void AMI over N, let s be a programmed finite partial state of S, and let f be a function from the instructions of S into the instructions of S. Then $f \cdot s$ is a programmed finite partial state of S.

In the sequel i is an instruction of $\mathbf{SCM}_{\rm FSA}$.

One can prove the following proposition

(23) $\mathrm{IncAddr}(\mathrm{IncAddr}(i,m),n) = \mathrm{IncAddr}(i,m+n)$.

4. Incremeting addresses in a finite partial state

Let p be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$ and let k be a natural number. The functor $\mathrm{IncAddr}(p,k)$ yields a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$ and is defined as follows:

(Def. 6) $\mathrm{dom}\,\mathrm{IncAddr}(p,k) = \mathrm{dom}\, p$ and for every m such that $\mathrm{insloc}(m) \in \mathrm{dom}\, p$ holds $(\mathrm{IncAddr}(p,k))(\mathrm{insloc}(m)) = \mathrm{IncAddr}(\pi_{\mathrm{insloc}(m)} p, k)$.

One can prove the following propositions:

(24) Let p be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$, k be a natural number, and l be an instruction-location of $\mathbf{SCM}_{\rm FSA}$. If $l \in \mathrm{dom}\, p$, then $(\mathrm{IncAddr}(p,k))(l) = \mathrm{IncAddr}(\pi_l p, k)$.

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

(26) Let f be a function from the instructions of $\mathbf{SCM}_{\rm FSA}$ into the instructions of $\mathbf{SCM}_{\rm FSA}$. Suppose $f = \mathrm{id}_{\text{the instructions of } \mathbf{SCM}_{\rm FSA}} +\cdot (\mathbf{halt}_{\mathbf{SCM}_{\rm FSA}} \dot{\longmapsto} i)$. Let s be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$. Then $\mathrm{IncAddr}(f \cdot s, n) = (\mathrm{id}_{\text{the instructions of } \mathbf{SCM}_{\rm FSA}} +\cdot (\mathbf{halt}_{\mathbf{SCM}_{\rm FSA}} \dot{\longmapsto} \mathrm{IncAddr}(i,n))) \cdot \mathrm{IncAddr}(s,n)$.

(27) For every programmed finite partial state I of $\mathbf{SCM}_{\rm FSA}$ holds $\mathrm{IncAddr}(\mathrm{IncAddr}(I,m),n) = \mathrm{IncAddr}(I,m+n)$.

(28) For every state s of $\mathbf{SCM}_{\rm FSA}$ holds $\mathrm{Exec}(\mathrm{IncAddr}(\mathrm{CurInstr}(s),k), s +\cdot \text{Start-At}(\mathbf{IC}_s + k)) = \mathrm{Following}(s) +\cdot \text{Start-At}(\mathbf{IC}_{\mathrm{Following}(s)} + k)$.

(29) Let I_2 be an instruction of $\mathbf{SCM}_{\rm FSA}$, s be a state of $\mathbf{SCM}_{\rm FSA}$, p be a finite partial state of $\mathbf{SCM}_{\rm FSA}$, and i, j, k be natural numbers. If $\mathbf{IC}_s = \mathrm{insloc}(j+k)$, then $\mathrm{Exec}(I_2, s +\cdot \text{Start-At}(\mathbf{IC}_s -' k)) = \mathrm{Exec}(\mathrm{IncAddr}(I_2,k), s) +\cdot \text{Start-At}(\mathbf{IC}_{\mathrm{Exec}(\mathrm{IncAddr}(I_2,k),s)} -' k)$.

5. SHIFTING THE FINITE PARTIAL STATE

Let p be a finite partial state of $\mathbf{SCM}_{\rm FSA}$ and let k be a natural number. The functor $\mathrm{Shift}(p,k)$ yields a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$ and is defined by:

(Def. 7) $\mathrm{dom}\,\mathrm{Shift}(p,k) = \{\mathrm{insloc}(m+k) : \mathrm{insloc}(m) \in \mathrm{dom}\,p\}$ and for every m such that $\mathrm{insloc}(m) \in \mathrm{dom}\,p$ holds $(\mathrm{Shift}(p,k))(\mathrm{insloc}(m+k)) = p(\mathrm{insloc}(m))$.

One can prove the following propositions:

(30) Let l be an instruction-location of $\mathbf{SCM}_{\rm FSA}$, k be a natural number, and p be a finite partial state of $\mathbf{SCM}_{\rm FSA}$. If $l \in \mathrm{dom}\,p$, then $(\mathrm{Shift}(p,k))(l+k) = p(l)$.

(31) Let p be a finite partial state of $\mathbf{SCM}_{\rm FSA}$ and k be a natural number. Then $\mathrm{dom}\,\mathrm{Shift}(p,k) = \{i_1+k; i_1$ ranges over instruction-locations of $\mathbf{SCM}_{\rm FSA}$: $i_1 \in \mathrm{dom}\,p\}$.

(32) For every finite partial state I of $\mathbf{SCM}_{\rm FSA}$ holds $\mathrm{Shift}(\mathrm{Shift}(I,m),n) = \mathrm{Shift}(I,m+n)$.

(33) Let s be a programmed finite partial state of $\mathbf{SCM}_{\rm FSA}$, f be a function from the instructions of $\mathbf{SCM}_{\rm FSA}$ into the instructions of $\mathbf{SCM}_{\rm FSA}$, and given n. Then $\mathrm{Shift}(f\cdot s,n) = f\cdot\mathrm{Shift}(s,n)$.

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

(35) For all natural numbers i, j and for every programmed finite partial state p of $\mathbf{SCM}_{\rm FSA}$ holds $\mathrm{Shift}(\mathrm{IncAddr}(p,i),j) = \mathrm{IncAddr}(\mathrm{Shift}(p,j),i)$.

REFERENCES

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

[2] 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.

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

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

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

[6] 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.

[7] 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.

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

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

[10] Takaya Nishiyama and Yasuho Mizuhara. Binary arithmetics. *Journal of Formalized Mathematics*, 5, 1993. http://mizar.org/JFM/Vol5/binarith.html.

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

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

[13] Yasushi Tanaka. Relocatability. *Journal of Formalized Mathematics*, 6, 1994. http://mizar.org/JFM/Vol6/reloc.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 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.

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

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

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

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

Received February 14, 1996

Published January 2, 2004