JOURNAL OF FORMALIZED MATHEMATICS
Volume 11, Released 1999, Published 2003
Inst. of Computer Science, Univ. of Białystok

Computation and Program Shift in the SCMPDS Computer[1]

Jing-Chao Chen
Shanghai Jiaotong University

Summary. A finite partial state is said to be autonomic if the computation results in any two states containing it are same on its domain. On the basis of this definition, this article presents some computation results about autonomic finite partial states of the SCMPDS computer. Because the instructions of the SCMPDS computer are more complicated than those of the SCMFSA computer, the results given by this article are weaker than those reported previously by the article on the SCMFSA computer. The second task of this article is to define the notion of program shift. The importance of this notion is that the computation of some program blocks can be simplified by shifting a program block to the initial position.

MML Identifier: `SCMPDS_3`.

WWW: `http://mizar.org/JFM/Vol11/scmpds_3.html`

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

1. PRELIMINARIES

In this paper k, m, n denote natural numbers.

The following propositions are true:

(1) Let n be a natural number. Suppose $n \leq 13$. Then $n = 0$ or $n = 1$ or $n = 2$ or $n = 3$ or $n = 4$ or $n = 5$ or $n = 6$ or $n = 7$ or $n = 8$ or $n = 9$ or $n = 10$ or $n = 11$ or $n = 12$ or $n = 13$.

(2) For every integer k_1 and for all states s_1, s_2 of SCMPDS such that $\mathbf{IC}_{(s_1)} = \mathbf{IC}_{(s_2)}$ holds $\mathrm{ICplusConst}(s_1, k_1) = \mathrm{ICplusConst}(s_2, k_1)$.

(3) Let k_1 be an integer, a be an Int position, and s_1, s_2 be states of SCMPDS. If $s_1 \restriction \text{Data-Loc}_{\mathrm{SCM}} = s_2 \restriction \text{Data-Loc}_{\mathrm{SCM}}$, then $s_1(\mathrm{DataLoc}(s_1(a), k_1)) = s_2(\mathrm{DataLoc}(s_2(a), k_1))$.

(4) For every Int position a and for all states s_1, s_2 of SCMPDS such that $s_1 \restriction \text{Data-Loc}_{\mathrm{SCM}} = s_2 \restriction \text{Data-Loc}_{\mathrm{SCM}}$ holds $s_1(a) = s_2(a)$.

(5) The carrier of SCMPDS $= \{\mathbf{IC}_{\mathrm{SCMPDS}}\} \cup \text{Data-Loc}_{\mathrm{SCM}} \cup$ the instruction locations of SCMPDS.

(6) $\mathbf{IC}_{\mathrm{SCMPDS}} \notin \text{Data-Loc}_{\mathrm{SCM}}$.

(7) For all states s_1, s_2 of SCMPDS such that $s_1 \restriction (\text{Data-Loc}_{\mathrm{SCM}} \cup \{\mathbf{IC}_{\mathrm{SCMPDS}}\}) = s_2 \restriction (\text{Data-Loc}_{\mathrm{SCM}} \cup \{\mathbf{IC}_{\mathrm{SCMPDS}}\})$ and for every instruction l of SCMPDS holds $\mathrm{Exec}(l, s_1) \restriction (\text{Data-Loc}_{\mathrm{SCM}} \cup \{\mathbf{IC}_{\mathrm{SCMPDS}}\}) = \mathrm{Exec}(l, s_2) \restriction (\text{Data-Loc}_{\mathrm{SCM}} \cup \{\mathbf{IC}_{\mathrm{SCMPDS}}\})$.

[1] This work was done while the author visited Shinshu University March–April 1999.

(8) For every instruction i of SCMPDS and for every state s of SCMPDS holds $\mathrm{Exec}(i,s){\restriction}\mathrm{Instr\text{-}Loc}_{\mathrm{SCM}} = s{\restriction}\mathrm{Instr\text{-}Loc}_{\mathrm{SCM}}$.

2. FINITE PARTIAL STATES OF SCMPDS

The following propositions are true:

(9) For every finite partial state p of SCMPDS holds $\mathrm{DataPart}(p) = p{\restriction}\mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$.

(10) For every finite partial state p of SCMPDS holds p is data-only iff $\mathrm{dom}\, p \subseteq \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$.

(11) For every finite partial state p of SCMPDS holds $\mathrm{dom}\,\mathrm{DataPart}(p) \subseteq \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$.

(12) For every finite partial state p of SCMPDS holds $\mathrm{dom}\,\mathrm{ProgramPart}(p) \subseteq$ the instruction locations of SCMPDS.

(13) Let i be an instruction of SCMPDS, s be a state of SCMPDS, and p be a programmed finite partial state of SCMPDS. Then $\mathrm{Exec}(i, s{+\cdot}p) = \mathrm{Exec}(i,s){+\cdot}p$.

(14) For every state s of SCMPDS and for every instruction-location i_1 of SCMPDS and for every Int position a holds $s(a) = (s{+\cdot}\,\mathrm{Start\text{-}At}(i_1))(a)$.

(15) For all states s, t of SCMPDS holds $s{+\cdot}t{\restriction}\mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$ is a state of SCMPDS.

3. AUTONOMIC FINITE PARTIAL STATES OF SCMPDS AND ITS COMPUTATION

Let l_1 be an Int position and let a be an integer. Then $l_1 \longmapsto a$ is a finite partial state of SCMPDS.

We now state the proposition

(16) For every autonomic finite partial state p of SCMPDS such that $\mathrm{DataPart}(p) \neq \emptyset$ holds $\mathbf{IC}_{\mathrm{SCMPDS}} \in \mathrm{dom}\, p$.

Let us note that there exists a finite partial state of SCMPDS which is autonomic and non programmed.

One can prove the following propositions:

(17) For every autonomic non programmed finite partial state p of SCMPDS holds $\mathbf{IC}_{\mathrm{SCMPDS}} \in \mathrm{dom}\, p$.

(18) Let s_1, s_2 be states of SCMPDS and k_1, k_2, m be integers. If $\mathbf{IC}_{(s_1)} = \mathbf{IC}_{(s_2)}$ and $k_1 \neq k_2$ and $m = \mathbf{IC}_{(s_1)}$ and $(m-2)+2\cdot k_1 \geq 0$ and $(m-2)+2\cdot k_2 \geq 0$, then $\mathrm{ICplusConst}(s_1,k_1) \neq \mathrm{ICplusConst}(s_2,k_2)$.

(19) For all states s_1, s_2 of SCMPDS and for all natural numbers k_1, k_2 such that $\mathbf{IC}_{(s_1)} = \mathbf{IC}_{(s_2)}$ and $k_1 \neq k_2$ holds $\mathrm{ICplusConst}(s_1,k_1) \neq \mathrm{ICplusConst}(s_2,k_2)$.

(20) For every state s of SCMPDS holds $\mathrm{Next}(\mathbf{IC}_s) = \mathrm{ICplusConst}(s,1)$.

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

(22) Let p be an autonomic non programmed finite partial state of SCMPDS and s be a state of SCMPDS. If $p \subseteq s$, then for every natural number i holds $\mathbf{IC}_{(\mathrm{Computation}(s))(i)} \in \mathrm{dom}\,\mathrm{ProgramPart}(p)$.

(23) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i be a natural number. Then $\mathbf{IC}_{(\mathrm{Computation}(s_1))(i)} = \mathbf{IC}_{(\mathrm{Computation}(s_2))(i)}$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = \mathrm{CurInstr}((\mathrm{Computation}(s_2))(i))$.

(24) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i be a natural number, k_1, k_2 be integers, and a, b be Int positions. Suppose $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = (a,k_1) := (b,k_2)$ and $a \in \mathrm{dom}\, p$ and $\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1) \in \mathrm{dom}\, p$. Then $(\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(b),k_2)) = (\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))(i)(b),$

(25) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i be a natural number, k_1, k_2 be integers, and a, b be Int positions. Suppose $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = \mathrm{AddTo}(a,k_1,b,k_2)$ and $a \in \mathrm{dom}\, p$ and $\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1) \in \mathrm{dom}\, p$. Then $(\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(b),k_2)) = (\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))(i)(b),$

(26) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i be a natural number, k_1, k_2 be integers, and a, b be Int positions. Suppose $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = \mathrm{SubFrom}(a,k_1,b,k_2)$ and $a \in \mathrm{dom}\, p$ and $\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1) \in \mathrm{dom}\, p$. Then $(\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(b),k_2)) = (\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))($

(27) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i be a natural number, k_1, k_2 be integers, and a, b be Int positions. Suppose $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = \mathrm{MultBy}(a,k_1,b,k_2)$ and $a \in \mathrm{dom}\, p$ and $\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1) \in \mathrm{dom}\, p$. Then $(\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1)) \cdot (\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(b),k$
$(\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))(i)(a),k_1)) \cdot (\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))(i)(b),k$

(28) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i, m be natural numbers, a be an Int position, and k_1, k_2 be integers. Suppose $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = (a,k_1) <> 0\_\mathrm{goto}\, k_2$ and $m = \mathbf{IC}_{(\mathrm{Computation}(s_1))(i)}$ and $(m-2)+2 \cdot k_2 \geq 0$ and $k_2 \neq 1$. Then $(\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1)) = 0$ if and only if $(\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))(i)(a),k_1)) = 0$.

(29) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i, m be natural numbers, a be an Int position, and k_1, k_2 be integers. Suppose $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = (a,k_1) <= 0\_\mathrm{goto}\, k_2$ and $m = \mathbf{IC}_{(\mathrm{Computation}(s_1))(i)}$ and $(m-2)+2 \cdot k_2 \geq 0$ and $k_2 \neq 1$. Then $(\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1)) > 0$ if and only if $(\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))(i)(a),k_1)) > 0$.

(30) Let p be an autonomic non programmed finite partial state of SCMPDS and s_1, s_2 be states of SCMPDS. Suppose $p \subseteq s_1$ and $p \subseteq s_2$. Let i, m be natural numbers, a be an Int position, and k_1, k_2 be integers. Suppose $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)) = (a,k_1) >= 0\_\mathrm{goto}\, k_2$ and $m = \mathbf{IC}_{(\mathrm{Computation}(s_1))(i)}$ and $(m-2)+2 \cdot k_2 \geq 0$ and $k_2 \neq 1$. Then $(\mathrm{Computation}(s_1))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_1))(i)(a),k_1)) < 0$ if and only if $(\mathrm{Computation}(s_2))(i)(\mathrm{DataLoc}((\mathrm{Computation}(s_2))(i)(a),k_1)) < 0$.

4. PROGRAM SHIFT IN THE SCMPDS COMPUTER

Let us consider k. The functor $\mathrm{inspos}\, k$ yields an instruction-location of SCMPDS and is defined by:

(Def. 2)[1] $\mathrm{inspos}\, k = \mathbf{i}_k$.

One can prove the following propositions:

(31) For all natural numbers k_1, k_2 such that $k_1 \neq k_2$ holds $\mathrm{inspos}\, k_1 \neq \mathrm{inspos}\, k_2$.

(32) For every instruction-location i_2 of SCMPDS there exists a natural number i such that $i_2 = \mathrm{inspos}\, i$.

[1] The definition (Def. 1) has been removed.

Let l_2 be an instruction-location of SCMPDS and let k be a natural number. The functor $l_2 + k$ yielding an instruction-location of SCMPDS is defined by:

(Def. 3) There exists a natural number m such that $l_2 = \mathrm{inspos}\, m$ and $l_2 + k = \mathrm{inspos}\, m + k$.

The functor $l_2 -' k$ yields an instruction-location of SCMPDS and is defined as follows:

(Def. 4) There exists a natural number m such that $l_2 = \mathrm{inspos}\, m$ and $l_2 -' k = \mathrm{inspos}\, m -' k$.

The following propositions are true:

(33) For every instruction-location l of SCMPDS and for all m, n holds $(l+m)+n = l+(m+n)$.

(34) For every instruction-location l_2 of SCMPDS and for every natural number k holds $(l_2 + k) -' k = l_2$.

(35) For all instruction-locations l_3, l_4 of SCMPDS and for every natural number k holds $\text{Start-At}(l_3 + k) = \text{Start-At}(l_4 + k)$ iff $\text{Start-At}(l_3) = \text{Start-At}(l_4)$.

(36) For all instruction-locations l_3, l_4 of SCMPDS and for every natural number k such that $\text{Start-At}(l_3) = \text{Start-At}(l_4)$ holds $\text{Start-At}(l_3 -' k) = \text{Start-At}(l_4 -' k)$.

Let I_1 be a finite partial state of SCMPDS. We say that I_1 is initial if and only if:

(Def. 5) For all m, n such that $\mathrm{inspos}\, n \in \mathrm{dom}\, I_1$ and $m < n$ holds $\mathrm{inspos}\, m \in \mathrm{dom}\, I_1$.

The finite partial state SCMPDS − Stop of SCMPDS is defined as follows:

(Def. 6) $\mathrm{SCMPDS} - \mathrm{Stop} = \mathrm{inspos}\, 0 \longmapsto \mathbf{halt}_{\mathrm{SCMPDS}}$.

Let us mention that SCMPDS − Stop is non empty, initial, and programmed.

One can check that there exists a finite partial state of SCMPDS which is initial, programmed, and non empty.

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

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

The following propositions are true:

(37) Let l be an instruction-location of SCMPDS, k be a natural number, and p be a programmed finite partial state of SCMPDS. If $l \in \mathrm{dom}\, p$, then $(\mathrm{Shift}(p,k))(l+k) = p(l)$.

(38) Let p be a programmed finite partial state of SCMPDS and k be a natural number. Then $\mathrm{dom}\,\mathrm{Shift}(p,k) = \{i_2 + k; i_2$ ranges over instruction-locations of SCMPDS: $i_2 \in \mathrm{dom}\, p\}$.

(39) For every programmed finite partial state I of SCMPDS holds $\mathrm{Shift}(\mathrm{Shift}(I,m),n) = \mathrm{Shift}(I,m+n)$.

(40) Let s be a programmed finite partial state of SCMPDS, f be a function from the instructions of SCMPDS into the instructions of SCMPDS, and given n. Then $\mathrm{Shift}(f \cdot s,n) = f \cdot \mathrm{Shift}(s,n)$.

(41) For all programmed finite partial states I, J of SCMPDS holds $\mathrm{Shift}(I{+\cdot}J,n) = \mathrm{Shift}(I,n){+\cdot}\,\mathrm{Shift}(J,n)$.

ACKNOWLEDGMENTS

We wish to thank Prof. Y. Nakamura for many helpful suggestions.

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. The ordinal numbers. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/ordinal1.html.

[3] Grzegorz Bancerek. Sequences of ordinal numbers. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/ordinal2.html.

[4] Grzegorz Bancerek. König's theorem. *Journal of Formalized Mathematics*, 2, 1990. http://mizar.org/JFM/Vol2/card_3.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. 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.

[9] Jing-Chao Chen. The SCMPDS computer and the basic semantics of its instructions. *Journal of Formalized Mathematics*, 11, 1999. http://mizar.org/JFM/Vol11/scmpds_2.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] Yatsuka Nakamura and Andrzej Trybulec. On a mathematical model of programs. *Journal of Formalized Mathematics*, 4, 1992. http://mizar.org/JFM/Vol4/ami_2.html.

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

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

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

[19] Wojciech A. Trybulec. Groups. *Journal of Formalized Mathematics*, 2, 1990. http://mizar.org/JFM/Vol2/group_1.html.

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

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

Received June 15, 1999

Published January 2, 2004