JOURNAL OF FORMALIZED MATHEMATICS
Volume 10, Released 1998, Published 2003
Inst. of Computer Science, Univ. of Białystok

Bubble Sort on $\mathbf{SCM}_{\mathrm{FSA}}$

Jing-Chao Chen
Shanghai Jiaotong University

Yatsuka Nakamura
Shinshu University
Nagano

Summary. We present the bubble sorting algorithm using macro instructions such as the if-Macro (conditional branch macro instructions) and the Times-Macro (for-loop macro instructions) etc. The correctness proof of the program should include the proof of autonomic, halting and the correctness of the program result. In the three terms, we justify rigorously the correctness of the bubble sorting algorithm. In order to prove it is autonomic, we use the following theorem: if all variables used by the program are initialized, it is autonomic. This justification method probably reveals that autonomic concept is not important.

MML Identifier: `SCMBSORT`.

WWW: `http://mizar.org/JFM/Vol10/scmbsort.html`

The articles [26], [25], [35], [9], [10], [27], [8], [33], [36], [22], [11], [13], [14], [18], [17], [34], [12], [21], [29], [24], [32], [16], [7], [30], [28], [15], [5], [6], [31], [23], [1], [2], [4], [20], [3], and [19] provide the notation and terminology for this paper.

For simplicity, we adopt the following convention: p is a programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$, i_1 is an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$, i, j, k are natural numbers, f_1, f are finite sequence locations, a, b are integer locations, and l_1 is an instruction-location of $\mathbf{SCM}_{\mathrm{FSA}}$.

Next we state a number of propositions:

(3)[1] Let I be a macro instruction and a, b be integer locations. If I does not destroy b and $a \neq b$, then $\mathrm{Times}(a, I)$ does not destroy b.

(4) For every function f and for all sets a, b, n, m holds $(f+\cdot(a \dot{\longmapsto} b)+\cdot(m \dot{\longmapsto} n))(m) = n$.

(5) For every function f and for all sets n, m holds $(f+\cdot(n \dot{\longmapsto} m)+\cdot(m \dot{\longmapsto} n))(n) = m$.

(6) For every function f and for all sets a, b, n, m, x such that $x \neq m$ and $x \neq a$ holds $(f+\cdot(a \dot{\longmapsto} b)+\cdot(m \dot{\longmapsto} n))(x) = f(x)$.

(7) Let f, g be functions and m, n be sets. Suppose that

(i) $f(m) = g(n)$,

(ii) $f(n) = g(m)$,

(iii) $m \in \mathrm{dom}\, f$,

(iv) $n \in \mathrm{dom}\, f$,

(v) $\mathrm{dom}\, f = \mathrm{dom}\, g$, and

(vi) for every set k such that $k \neq m$ and $k \neq n$ and $k \in \mathrm{dom}\, f$ holds $f(k) = g(k)$.

Then f and g are fiberwise equipotent.

[1] The propositions (1) and (2) have been removed.

(8) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, f be a finite sequence location, and a, b be integer locations. Then $(\mathrm{Exec}(b{:=}f_a, s))(b) = s(f)_{|s(a)|}$.

(9) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, f be a finite sequence location, and a, b be integer locations. Then $(\mathrm{Exec}(f_a{:=}b, s))(f) = s(f) +\cdot (|s(a)|, s(b))$.

(10) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, f be a finite sequence location, m, n be natural numbers, and a be an integer location. If $m \neq n+1$, then $(\mathrm{Exec}(\mathrm{intloc}(m){:=}f_a, \mathrm{Initialize}(s)))(\mathrm{intloc}(n+1)) = s(\mathrm{intloc}(n+1))$.

(11) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, m, n be natural numbers, and a be an integer location. If $m \neq n+1$, then $(\mathrm{Exec}(\mathrm{intloc}(m){:=}a, \mathrm{Initialize}(s)))(\mathrm{intloc}(n+1)) = s(\mathrm{intloc}(n+1))$.

(12) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, f be a finite sequence location, and a be a read-write integer location. Then $(\mathrm{IExec}(\mathrm{Stop}_{\mathbf{SCM}_{\mathrm{FSA}}}, s))(a) = s(a)$ and $(\mathrm{IExec}(\mathrm{Stop}_{\mathbf{SCM}_{\mathrm{FSA}}}, s))(f) = s(f)$.

In the sequel n denotes a natural number.

We now state a number of propositions:

(13) If $n \leq 10$, 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 $n = 10$.

(14) Suppose $n \leq 12$. 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 $n = 12$.

(15) Let f, g be functions and X be a set. If $\mathrm{dom} f = \mathrm{dom} g$ and for every set x such that $x \in X$ holds $f(x) = g(x)$, then $f{\restriction}X = g{\restriction}X$.

(16) If $i_1 \in \mathrm{rng}\, p$ and if $i_1 = a{:=}b$ or $i_1 = \mathrm{AddTo}(a,b)$ or $i_1 = \mathrm{SubFrom}(a,b)$ or $i_1 = \mathrm{MultBy}(a,b)$ or $i_1 = \mathrm{Divide}(a,b)$, then $a \in \mathrm{UsedIntLoc}(p)$ and $b \in \mathrm{UsedIntLoc}(p)$.

(17) If $i_1 \in \mathrm{rng}\, p$ and if $i_1 = \mathbf{if}\ a = 0\ \mathbf{goto}\ l_1$ or $i_1 = \mathbf{if}\ a > 0\ \mathbf{goto}\ l_1$, then $a \in \mathrm{UsedIntLoc}(p)$.

(18) If $i_1 \in \mathrm{rng}\, p$ and if $i_1 = b{:=}{f_1}_a$ or $i_1 = {f_1}_a{:=}b$, then $a \in \mathrm{UsedIntLoc}(p)$ and $b \in \mathrm{UsedIntLoc}(p)$.

(19) If $i_1 \in \mathrm{rng}\, p$ and if $i_1 = b{:=}{f_1}_a$ or $i_1 = {f_1}_a{:=}b$, then $f_1 \in \mathrm{UsedInt}^*\mathrm{Loc}(p)$.

(20) If $i_1 \in \mathrm{rng}\, p$ and if $i_1 = a{:=}\mathrm{len} f_1$ or $i_1 = f_1{:=}\langle \underbrace{0, \ldots, 0}_{a} \rangle$, then $a \in \mathrm{UsedIntLoc}(p)$.

(21) If $i_1 \in \mathrm{rng}\, p$ and if $i_1 = a{:=}\mathrm{len} f_1$ or $i_1 = f_1{:=}\langle \underbrace{0, \ldots, 0}_{a} \rangle$, then $f_1 \in \mathrm{UsedInt}^*\mathrm{Loc}(p)$.

(22) Let N be a set with non empty elements, S be a steady-programmed IC-Ins-separated definite non empty non void AMI over N, p be a programmed finite partial state of S, and s_1, s_2 be states of S. If $p \subseteq s_1$ and $p \subseteq s_2$, then $(\mathrm{Computation}(s_1))(i){\restriction}\mathrm{dom}\, p = (\mathrm{Computation}(s_2))(i){\restriction}\mathrm{dom}\, p$.

(23) Let t be a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$, p be a macro instruction, and x be a set. Suppose $\mathrm{dom}\, t \subseteq \text{Int-Locations} \cup \text{FinSeq-Locations}$ and $x \in \mathrm{dom}\, t \cup \mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p)$. Then x is an integer location and a finite sequence location.

(25)[2] Let i, k be natural numbers, t be a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$, p be a macro instruction, and s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose that $k \leq i$ and $p \subseteq s_1$ and $p \subseteq s_2$ and $\mathrm{dom}\, t \subseteq \text{Int-Locations} \cup \text{FinSeq-Locations}$ and for every j holds $\mathbf{IC}_{(\mathrm{Computation}(s_1))(j)} \in \mathrm{dom}\, p$ and $\mathbf{IC}_{(\mathrm{Computation}(s_2))(j)} \in \mathrm{dom}\, p$ and $(\mathrm{Computation}(s_1))(k)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}) = (\mathrm{Computation}(s_2))(k)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}})$ and $(\mathrm{Computation}(s_1))(k){\restriction}(\mathrm{dom}\, t \cup \mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p)) = (\mathrm{Computation}(s_2))(k){\restriction}(\mathrm{dom}\, t \cup \mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p))$. Then $(\mathrm{Computation}(s_1))(i)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}) = (\mathrm{Computation}(s_2))(i)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}})$ and $(\mathrm{Computation}(s_1))(i){\restriction}(\mathrm{dom}\, t \cup \mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p)) = (\mathrm{Computation}(s_2))(i){\restriction}(\mathrm{dom}\, t \cup \mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p))$.

[2] The proposition (24) has been removed.

(26) Let i, k be natural numbers, p be a macro instruction, and s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose $k \leq i$ and $p \subseteq s_1$ and $p \subseteq s_2$ and for every j holds $\mathbf{IC}_{(\mathrm{Computation}(s_1))(j)} \in \mathrm{dom}\, p$ and $\mathbf{IC}_{(\mathrm{Computation}(s_2))(j)} \in \mathrm{dom}\, p$ and $(\mathrm{Computation}(s_1))(k)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}) = (\mathrm{Computation}(s_2))(k)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}})$ and $(\mathrm{Computation}(s_1))(k) \restriction (\mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p)) = (\mathrm{Computation}(s_2))(k) \restriction (\mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p))$. Then $(\mathrm{Computation}(s_1))(i)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}) = (\mathrm{Computation}(s_2))(i)(\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}})$ and $(\mathrm{Computation}(s_1))(i) \restriction (\mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p)) = (\mathrm{Computation}(s_2))(i) \restriction (\mathrm{UsedInt}^*\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p))$.

(29)[3] For all macro instructions I, J and for every integer location a holds $\mathrm{UsedIntLoc}(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J) = \{a\} \cup \mathrm{UsedIntLoc}(I) \cup \mathrm{UsedIntLoc}(J)$ and $\mathrm{UsedIntLoc}(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J) = \{a\} \cup \mathrm{UsedIntLoc}(I) \cup \mathrm{UsedIntLoc}(J)$.

(30) For every macro instruction I and for every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{UsedIntLoc}(\mathrm{Directed}(I,l)) = \mathrm{UsedIntLoc}(I)$.

(31) For every integer location a and for every macro instruction I holds $\mathrm{UsedIntLoc}(\mathrm{Times}(a,I)) = \mathrm{UsedIntLoc}(I) \cup \{a, \mathrm{intloc}(0)\}$.

(32) For all sets x_1, x_2, x_3 holds $\{x_2, x_1\} \cup \{x_3, x_1\} = \{x_1, x_2, x_3\}$.

(35)[4] For all macro instructions I, J and for every integer location a holds $\mathrm{UsedInt}^*\mathrm{Loc}(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J) = \mathrm{UsedInt}^*\mathrm{Loc}(I) \cup \mathrm{UsedInt}^*\mathrm{Loc}(J)$ and $\mathrm{UsedInt}^*\mathrm{Loc}(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J) = \mathrm{UsedInt}^*\mathrm{Loc}(I) \cup \mathrm{UsedInt}^*\mathrm{Loc}(J)$.

(36) For every macro instruction I and for every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{UsedInt}^*\mathrm{Loc}(\mathrm{Directed}(I,l)) = \mathrm{UsedInt}^*\mathrm{Loc}(I)$.

(37) For every integer location a and for every macro instruction I holds $\mathrm{UsedInt}^*\mathrm{Loc}(\mathrm{Times}(a,I)) = \mathrm{UsedInt}^*\mathrm{Loc}(I)$.

Let f be a finite sequence location and let t be a finite sequence of elements of $\mathbb{Z}$. Then $f \dot{\longmapsto} t$ is a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$.

One can prove the following propositions:

(38) Every finite sequence of elements of $\mathbb{Z}$ is a finite sequence of elements of $\mathbb{R}$.

(39) Let t be a finite sequence of elements of $\mathbb{Z}$. Then there exists a finite sequence u of elements of $\mathbb{R}$ such that t and u are fiberwise equipotent and u is a finite sequence of elements of $\mathbb{Z}$ and non-increasing.

(40) $\mathrm{dom}((\mathrm{intloc}(0) \dot{\longmapsto} 1) {+}\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0))) = \{\mathrm{intloc}(0), \mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}\}$.

(41) For every macro instruction I holds $\mathrm{dom}\,\mathrm{Initialized}(I) = \mathrm{dom}\, I \cup \{\mathrm{intloc}(0), \mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}\}$.

(42) Let w be a finite sequence of elements of $\mathbb{Z}$, f be a finite sequence location, and I be a macro instruction. Then $\mathrm{dom}(\mathrm{Initialized}(I) {+}\cdot (f \dot{\longmapsto} w)) = \mathrm{dom}\, I \cup \{\mathrm{intloc}(0), \mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}, f\}$.

(43) For every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}} \neq l$.

(44) For every integer location a and for every macro instruction I holds $\mathrm{card}\,\mathrm{Times}(a,I) = \mathrm{card}\, I + 12$.

(45) For all instructions i_2, i_3, i_4 of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{card}(i_2;\ i_3;\ i_4) = 6$.

(46) Let t be a finite sequence of elements of $\mathbb{Z}$, f be a finite sequence location, and I be a macro instruction. Then $\mathrm{dom}\,\mathrm{Initialized}(I)$ misses $\mathrm{dom}(f \dot{\longmapsto} t)$.

(47) Let w be a finite sequence of elements of $\mathbb{Z}$, f be a finite sequence location, and I be a macro instruction. Then $\mathrm{Initialized}(I) {+}\cdot (f \dot{\longmapsto} w)$ starts at $\mathrm{insloc}(0)$.

[3] The propositions (27) and (28) have been removed.
[4] The propositions (33) and (34) have been removed.

(48) Let I, J be macro instructions, k be a natural number, and i be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$. If $k < \mathrm{card}\,J$ and $i = J(\mathrm{insloc}(k))$, then $(I;\,J)(\mathrm{insloc}(\mathrm{card}\,I + k)) = \mathrm{IncAddr}(i, \mathrm{card}\,I)$.

(49) Suppose that $i_1 = a{:=}b$ or $i_1 = \mathrm{AddTo}(a,b)$ or $i_1 = \mathrm{SubFrom}(a,b)$ or $i_1 = \mathrm{MultBy}(a,b)$ or $i_1 = \mathrm{Divide}(a,b)$ or $i_1 = \mathrm{goto}\ l_1$ or $i_1 = \mathbf{if}\ a = 0\ \mathbf{goto}\ l_1$ or $i_1 = \mathbf{if}\ a > 0\ \mathbf{goto}\ l_1$ or $i_1 = b{:=}f_a$ or $i_1 = f_a{:=}b$ or $i_1 = a{:=}\mathrm{len}\,f$ or $i_1 = f{:=}\langle\underbrace{0,\ldots,0}_{a}\rangle$. Then $i_1 \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(50) Let I, J be macro instructions, k be a natural number, and i be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose for every natural number n holds $\mathrm{IncAddr}(i,n) = i$ and $i \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$ and $k = \mathrm{card}\,I$. Then $(I;\,i;\,J)(\mathrm{insloc}(k)) = i$ and $(I;\,i;\,J)(\mathrm{insloc}(k+1)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I + 2)$.

(51) Let I, J be macro instructions and k be a natural number. If $k = \mathrm{card}\,I$, then $(I;\,(a{:=}b);\,J)(\mathrm{insloc}(k)) = a{:=}b$ and $(I;\,(a{:=}b);\,J)(\mathrm{insloc}(k+1)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I + 2)$.

(52) Let I, J be macro instructions and k be a natural number. If $k = \mathrm{card}\,I$, then $(I;\,(a{:=}\mathrm{len}\,f);\,J)(\mathrm{insloc}(k)) = a{:=}\mathrm{len}\,f$ and $(I;\,(a{:=}\mathrm{len}\,f);\,J)(\mathrm{insloc}(k+1)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I + 2)$.

(53) Let w be a finite sequence of elements of $\mathbb{Z}$, f be a finite sequence location, s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, and I be a macro instruction. If $\mathrm{Initialized}(I) {+\cdot} (f \dot{\longmapsto} w) \subseteq s$, then $I \subseteq s$.

(54) Let w be a finite sequence of elements of $\mathbb{Z}$, f be a finite sequence location, s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, and I be a macro instruction. If $\mathrm{Initialized}(I) {+\cdot} (f \dot{\longmapsto} w) \subseteq s$, then $s(f) = w$ and $s(\mathrm{intloc}(0)) = 1$.

(55) For every finite sequence location f and for every integer location a and for every state s of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\{a, \mathbf{IC}_{\mathbf{SCM}_{\mathrm{FSA}}}, f\} \subseteq \mathrm{dom}\,s$.

(56) For every macro instruction p and for every state s of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{UsedInt}^*\,\mathrm{Loc}(p) \cup \mathrm{UsedIntLoc}(p) \subseteq \mathrm{dom}\,s$.

(57) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, I be a macro instruction, and f be a finite sequence location. Then $(\mathrm{Result}(s {+\cdot}\, \mathrm{Initialized}(I)))(f) = (\mathrm{IExec}(I,s))(f)$.

Let f be a finite sequence location. The functor bubble-sort(f) yielding a macro instruction is defined by:

(Def. 1) $\text{bubble-sort}(f) = i_5;\ (a_1{:=}\mathrm{len}\,f);\ \mathrm{Times}(a_1, (a_2{:=}a_1);\ \mathrm{SubFrom}(a_2,a_0);\ (a_3{:=}\mathrm{len}\,f);\ \mathrm{Times}(a_2, (a_4{:=}a_3);\ \mathrm{SubFrom}(a_3,$ $0\ \mathbf{then}\ (a_6{:=}f_{a_4});\ (f_{a_3}{:=}a_6);\ (f_{a_4}{:=}a_5)\ \mathbf{else}\ (\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}})))))$, where $i_5 = (a_2{:=}a_0);\ (a_3{:=}a_0);\ (a_4{:=}a_0);\ (a_5{:=}a_0);\ (a_6{:=}a$ $a_2 = \mathrm{intloc}(2)$, $a_0 = \mathrm{intloc}(0)$, $a_3 = \mathrm{intloc}(3)$, $a_4 = \mathrm{intloc}(4)$, $a_5 = \mathrm{intloc}(5)$, $a_6 = \mathrm{intloc}(6)$, and $a_1 = \mathrm{intloc}(1)$.

The macro instruction the bubble sort algorithm is defined by:

(Def. 2) The bubble sort algorithm $=$ bubble-sort(fsloc(0)).

Next we state two propositions:

(58) For every finite sequence location f holds $\mathrm{UsedIntLoc}(\text{bubble-sort}(f)) = \{a_0, a_1, a_2, a_3, a_4, a_5, a_6\}$, where $a_0 = \mathrm{intloc}(0)$, $a_1 = \mathrm{intloc}(1)$, $a_2 = \mathrm{intloc}(2)$, $a_3 = \mathrm{intloc}(3)$, $a_4 = \mathrm{intloc}(4)$, $a_5 = \mathrm{intloc}(5)$, and $a_6 = \mathrm{intloc}(6)$.

(59) For every finite sequence location f holds $\mathrm{UsedInt}^*\,\mathrm{Loc}(\text{bubble-sort}(f)) = \{f\}$.

The partial function Sorting-Function from $\mathrm{FinPartSt}(\mathbf{SCM}_{\mathrm{FSA}})$ to $\mathrm{FinPartSt}(\mathbf{SCM}_{\mathrm{FSA}})$ is defined by the condition (Def. 3).

(Def. 3) Let p, q be finite partial states of $\mathbf{SCM}_{\mathrm{FSA}}$. Then $\langle p, q\rangle \in$ Sorting-Function if and only if there exists a finite sequence t of elements of $\mathbb{Z}$ and there exists a finite sequence u of elements of $\mathbb{R}$ such that t and u are fiberwise equipotent and u is a finite sequence of elements of $\mathbb{Z}$ and non-increasing and $p = \mathrm{fsloc}(0) \dot{\longmapsto} t$ and $q = \mathrm{fsloc}(0) \dot{\longmapsto} u$.

The following propositions are true:

(60) For every set p holds $p \in \operatorname{dom}$ Sorting-Function iff there exists a finite sequence t of elements of $\mathbb{Z}$ such that $p = \mathrm{fsloc}(0) \dot{\longmapsto} t$.

(61) Let t be a finite sequence of elements of $\mathbb{Z}$. Then there exists a finite sequence u of elements of $\mathbb{R}$ such that

(i) t and u are fiberwise equipotent,

(ii) u is non-increasing and a finite sequence of elements of $\mathbb{Z}$, and

(iii) $(\text{Sorting-Function})(\mathrm{fsloc}(0) \dot{\longmapsto} t) = \mathrm{fsloc}(0) \dot{\longmapsto} u$.

(62) For every finite sequence location f holds card bubble-sort$(f) = 63$.

(63) For every finite sequence location f and for every natural number k such that $k < 63$ holds $\mathrm{insloc}(k) \in \operatorname{dom}$ bubble-sort(f).

(64) bubble-sort$(\mathrm{fsloc}(0))$ is keepInt0 1 and InitHalting.

(65) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Then

(i) $s(f_0)$ and $(\mathrm{IExec}(\text{bubble-sort}(f_0), s))(f_0)$ are fiberwise equipotent, and

(ii) for all natural numbers i, j such that $i \geq 1$ and $j \leq \operatorname{len} s(f_0)$ and $i < j$ and for all integers x_1, x_2 such that $x_1 = (\mathrm{IExec}(\text{bubble-sort}(f_0), s))(f_0)(i)$ and $x_2 = (\mathrm{IExec}(\text{bubble-sort}(f_0), s))(f_0)(j)$ holds $x_1 \geq x_2$,

where $f_0 = \mathrm{fsloc}(0)$.

(66) Let i be a natural number, s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, and w be a finite sequence of elements of $\mathbb{Z}$. Suppose Initialized(the bubble sort algorithm) $+\cdot (\mathrm{fsloc}(0) \dot{\longmapsto} w) \subseteq s$. Then $\mathbf{IC}_{(\mathrm{Computation}(s))(i)} \in \operatorname{dom}$(the bubble sort algorithm).

(67) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and t be a finite sequence of elements of $\mathbb{Z}$. Suppose Initialized(the bubble sort algorithm) $+\cdot (\mathrm{fsloc}(0) \dot{\longmapsto} t) \subseteq s$. Then there exists a finite sequence u of elements of $\mathbb{R}$ such that

(i) t and u are fiberwise equipotent,

(ii) u is non-increasing and a finite sequence of elements of $\mathbb{Z}$, and

(iii) $(\mathrm{Result}(s))(\mathrm{fsloc}(0)) = u$.

(68) For every finite sequence w of elements of $\mathbb{Z}$ holds Initialized(the bubble sort algorithm) $+\cdot (\mathrm{fsloc}(0) \dot{\longmapsto} w)$ is autonomic.

(69) Initialized(the bubble sort algorithm) computes Sorting-Function.

REFERENCES

[1] Noriko Asamoto. Conditional branch macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part I. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa8a.html.

[2] Noriko Asamoto. Conditional branch macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part II. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa8b.html.

[3] Noriko Asamoto. Constant assignment macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part II. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa7b.html.

[4] Noriko Asamoto. The `loop` and `times` macroinstruction for $\mathbf{SCM}_{\mathrm{FSA}}$. *Journal of Formalized Mathematics*, 9, 1997. http://mizar.org/JFM/Vol9/scmfsa8c.html.

[5] Noriko Asamoto, Yatsuka Nakamura, Piotr Rudnicki, and Andrzej Trybulec. On the composition of macro instructions. Part II. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa6b.html.

[6] Noriko Asamoto, Yatsuka Nakamura, Piotr Rudnicki, and Andrzej Trybulec. On the composition of macro instructions. Part III. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa6c.html.

[7] Grzegorz Bancerek. Cardinal numbers. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/card_1.html.

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

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

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

[11] 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.

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

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

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

[15] Czesław Byliński. Partial functions. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/partfun1.html.

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

[17] 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.

[18] 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.

[19] Jing-Chao Chen and Yatsuka Nakamura. Initialization halting concepts and their basic properties of $\mathbf{SCM}_{\mathrm{FSA}}$. *Journal of Formalized Mathematics*, 10, 1998. http://mizar.org/JFM/Vol10/scm_halt.html.

[20] Jarosław Kotowicz. Functions and finite sequences of real numbers. *Journal of Formalized Mathematics*, 5, 1993. http://mizar.org/JFM/Vol5/rfinseq.html.

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

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

[23] Piotr Rudnicki and Andrzej Trybulec. Memory handling for $\mathbf{SCM}_{\mathrm{FSA}}$. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/sf_mastr.html.

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

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

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

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

[28] Andrzej Trybulec and Agata Darmochwał. Boolean domains. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/finsub_1.html.

[29] 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.

[30] 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.

[31] Andrzej Trybulec, Yatsuka Nakamura, and Noriko Asamoto. On the compositions of macro instructions. Part I. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa6a.html.

[32] 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.

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

[34] Wojciech A. Trybulec. Pigeon hole principle. *Journal of Formalized Mathematics*, 2, 1990. http://mizar.org/JFM/Vol2/finseq_4.html.

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

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

Received June 17, 1998

Published January 2, 2004