FORMALIZED MATHEMATICS
Volume 4, Number 1, 1993
Université Catholique de Louvain

Two Programs for SCM. Part II - Programs [1]

Grzegorz Bancerek
Polish Academy of Sciences
Institute of Mathematics
Warsaw

Piotr Rudnicki
University of Alberta
Department of Computing Science
Edmonton

Summary. We prove the correctness of two short programs for the **SCM** machine: one computes Fibonacci numbers and the other computes the *fusc* function of Dijkstra [11]. The formal definitions of these functions can be found in [5]. We prove the total correctness of the programs in two ways: by conducting inductions on computations and inductions on input data. In addition we characterize the concrete complexity of the programs as defined in [4].

MML Identifier: FIB_FUSC.

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

The program computing Fib is a finite sequence of elements of the instructions of **SCM** and is defined as follows:

(Def.1) The program computing Fib = $\langle \mathbf{if}\ \mathbf{d}_1 > 0\ \mathbf{goto}\ \mathbf{i}_2 \rangle \frown \langle \mathbf{halt}_{\mathbf{SCM}} \rangle \frown \langle \mathbf{d}_3{:=}\mathbf{d}_0 \rangle \frown \langle \mathrm{SubFrom}(\mathbf{d}_1, \mathbf{d}_0) \rangle \frown \langle \mathbf{if}\ \mathbf{d}_1 = 0\ \mathbf{goto}\ \mathbf{i}_1 \rangle \frown \langle \mathbf{d}_4{:=}\mathbf{d}_2 \rangle \frown \langle \mathbf{d}_2{:=}\mathbf{d}_3 \rangle \frown \langle \mathrm{AddTo}(\mathbf{d}_3, \mathbf{d}_4) \rangle \frown \langle \mathrm{goto}\ (\mathbf{i}_3) \rangle$.

The following proposition is true

(1) Let N be a natural number and let s be a state with instruction counter on 0, with the program computing Fib located from 0, and $\langle +1 \rangle \frown \langle +N \rangle \frown \langle +0 \rangle \frown \langle +0 \rangle$ from 0. Then
  (i) s is halting,
  (ii) if $N = 0$, then the complexity of $s = 1$,
  (iii) if $N > 0$, then the complexity of $s = 6 \cdot N - 2$, and
  (iv) $(\mathrm{Result}(s))(\mathbf{d}_3) = \mathrm{Fib}(N)$.

[1] This work was partially supported by NSERC Grant OGP9207 while the first author visited University of Alberta, May-June 1993.

ISSN 0777-4028

Let i be an integer. The functor Fusc(i) yields a natural number and is defined as follows:

(Def.2) There exists a natural number n such that $i = n$ and Fusc(i) = Fusc(n) or i is not a natural number and Fusc(i) = 0.

Let a, n be natural numbers. Then a^n is an integer.

The program computing Fusc is a finite sequence of elements of the instructions of **SCM** and is defined by:

(Def.3) The program computing Fusc = $\langle$**if** $\mathbf{d}_1 = 0$ **goto** $\mathbf{i}_8\rangle \frown \langle \mathbf{d}_4{:=}\mathbf{d}_0\rangle \frown \langle$Divide$(\mathbf{d}_1, \mathbf{d}_4)\rangle \frown \langle$**if** $\mathbf{d}_4 = 0$ **goto** $\mathbf{i}_6\rangle \frown \langle$AddTo$(\mathbf{d}_3, \mathbf{d}_2)\rangle \frown \langle$goto $(\mathbf{i}_0)\rangle \frown \langle$AddTo$(\mathbf{d}_2, \mathbf{d}_3)\rangle \frown \langle$goto $(\mathbf{i}_0)\rangle \frown \langle \mathbf{halt}_{\mathbf{SCM}}\rangle$.

We now state several propositions:

(2) Let N be a natural number. Suppose $N > 0$. Let s be a state with instruction counter on 0, with the program computing Fusc located from 0, and $\langle+2\rangle \frown \langle+N\rangle \frown \langle+1\rangle \frown \langle+0\rangle$ from 0. Then s is halting and (Result(s))($\mathbf{d}_3$) = Fusc(N) and the complexity of $s = 6\cdot(\lfloor\log_2 N\rfloor+1)+1$.

(3) Let N be a natural number, and let k, F_1, F_2 be natural numbers, and let s be a state with instruction counter on 3, with the program computing Fib located from 0, and $\langle+1\rangle \frown \langle+N\rangle \frown \langle+F_1\rangle \frown \langle+F_2\rangle$ from 0. Suppose $N > 0$ and $F_1 = \mathrm{Fib}(k)$ and $F_2 = \mathrm{Fib}(k+1)$. Then

(i) s is halting,

(ii) the complexity of $s = 6 \cdot N - 4$, and

(iii) there exists a natural number m such that $m = (k + N) - 1$ and (Result(s))($\mathbf{d}_2$) = Fib(m) and (Result(s))($\mathbf{d}_3$) = Fib($m+1$).

(4) Let N be a natural number and let s be a state with instruction counter on 0, with the program computing Fib located from 0, and $\langle+1\rangle \frown \langle+N\rangle \frown \langle+0\rangle \frown \langle+0\rangle$ from 0. Then

(i) s is halting,

(ii) if $N = 0$, then the complexity of $s = 1$,

(iii) if $N > 0$, then the complexity of $s = 6 \cdot N - 2$, and

(iv) (Result(s))($\mathbf{d}_3$) = Fib(N).

(5) Let n be a natural number, and let N, A, B be natural numbers, and let s be a state with instruction counter on 0, with the program computing Fusc located from 0, and $\langle+2\rangle \frown \langle+n\rangle \frown \langle+A\rangle \frown \langle+B\rangle$ from 0. Suppose $N > 0$ and Fusc(N) = $A \cdot$ Fusc(n) + $B \cdot$ Fusc($n+1$). Then

(i) s is halting,

(ii) (Result(s))($\mathbf{d}_3$) = Fusc(N),

(iii) if $n = 0$, then the complexity of $s = 1$, and

(iv) if $n > 0$, then the complexity of $s = 6 \cdot (\lfloor\log_2 n\rfloor + 1) + 1$.

(6) Let N be a natural number. Suppose $N > 0$. Let s be a state with instruction counter on 0, with the program computing Fusc located from 0, and $\langle+2\rangle \frown \langle+N\rangle \frown \langle+1\rangle \frown \langle+0\rangle$ from 0. Then

(i) s is halting,

(ii) (Result(s))($\mathbf{d}_3$) = Fusc(N),

(iii) if $N = 0$, then the complexity of $s = 1$, and
(iv) if $N > 0$, then the complexity of $s = 6 \cdot (\lfloor \log_2 N \rfloor + 1) + 1$.

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 Piotr Rudnicki. Development of terminology for **SCM**. *Formalized Mathematics*, 4(**1**):61–67, 1993.
[5] Grzegorz Bancerek and Piotr Rudnicki. Two programs for **SCM**. Part I - preliminaries. *Formalized Mathematics*, 4(**1**):69–72, 1993.
[6] Czesław Byliński. Finite sequences and tuples of elements of a non-empty sets. *Formalized Mathematics*, 1(**3**):529–536, 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. Products and coproducts in categories. *Formalized Mathematics*, 2(**5**):701–709, 1991.
[10] Agata Darmochwał. Finite sets. *Formalized Mathematics*, 1(1):165–167, 1990.
[11] Edsger W. Dijkstra. *Selected Writings on Computing, a Personal Perspective.*
[12] Rafał Kwiatek. Factorial and Newton coefficients. *Formalized Mathematics*, 1(**5**):887–890, 1990.
[13] Rafał Kwiatek and Grzegorz Zwara. The divisibility of integers and integer relative primes. *Formalized Mathematics*, 1(**5**):829–832, 1990.
[14] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Formalized Mathematics*, 3(**2**):151–160, 1992.
[15] Yatsuka Nakamura and Andrzej Trybulec. On a mathematical model of programs. *Formalized Mathematics*, 3(**2**):241–250, 1992.
[16] Konrad Raczkowski and Andrzej Nędzusiak. Real exponents and logarithms. *Formalized Mathematics*, 2(**2**):213–216, 1991.
[17] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[18] Andrzej Trybulec. Tuples, projections and Cartesian products. *Formalized Mathematics*, 1(**1**):97–105, 1990.
[19] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Formalized Mathematics*, 4(**1**):51–56, 1993.
[20] Michał J. Trybulec. Integers. *Formalized Mathematics*, 1(**3**):501–505, 1990.
[21] Wojciech A. Trybulec. Pigeon hole principle. *Formalized Mathematics*, 1(**3**):575–579, 1990.

Received October 8, 1993