

Formal Verification of a Generic Iterative Algorithm

Artur Kornilowicz
Faculty of Computer Science
University of Białystok
Poland

Summary. In this paper we formally verify in the Mizar system the correctness of a parameterized algorithm using SCM (Simple Concrete Model). Generic properties of the algorithm are formulated and proved as Mizar schemes. In the last section two classical cases: the computation of the factorial of a natural number and the computation of the n -th triangular number are verified using the established framework.

MSC: 68Q25 68W40 68V15 68V20

Keywords: formal verification; generic algorithm; factorial; triangle number

MML identifier: AMI_8, version: 8.1.15 5.103.1512

INTRODUCTION

Formal verification is one of the fundamental areas of formal methods, providing mathematical guarantees that a program satisfies its specification. Unlike testing, which can only increase confidence in the correctness of an implementation, formal verification establishes correctness for all admissible inputs. Interactive theorem provers, such as Mizar [2, 3], play an important role in this process by enabling the formalization of mathematical concepts, computational models, and correctness proofs of algorithms.

The aim of this paper is to formally verify two classical iterative algorithms: the computation of the factorial of a natural number and the computation of the n -th triangular number. Rather than verifying these algorithms independently, we consider their common control structure, represented by the following generic program:

```

0) if b > 0 then goto 2
1) halt
2) OPER
3) b := b-a
4) if b = 0 then goto 1
5) goto 2

```

where a , b , and c represent data-locations and a stores 1 as a loop step, b stores an input value, and c stores the result value.

The program is parameterized by the instruction `OPER`, which specifies the operation performed during each iteration of the loop. Consequently, the same control-flow structure can be used to implement different algorithms by instantiating `OPER` with an appropriate arithmetic instruction. The algorithm is formalized in the Mizar system using the SCM (Simple Concrete Model) [4]. The program is represented as a finite sequence of SCM instructions by the definition of the function `ResultOfOne(OPER)`, whose only parameter is the instruction executed in the loop body. This formulation cleanly separates the properties that depend solely on the control structure from those determined by the particular arithmetic operation. The correctness proof is carried out within the framework of operational semantics [?], where the execution of a program is modeled as a sequence of state transitions.

The analysis focuses on the evolution of the instruction counter (`IC`) and the contents of the registers representing the program variables (`dl.0`, `dl.1`, and `dl.2`) throughout the execution. Properties that are independent of the concrete definition of `OPER` are formulated and proved as a collection of generic Mizar schemes. These reusable results capture the behavior of the program's control flow and provide a common basis for verifying different instances of the algorithm.

In the final section of the paper, the generic schemes are instantiated to verify two concrete algorithms. In the first case, `OPER` is defined as the multiplication instruction `MultBy(dl.2,dl.1)`, yielding a formal proof of correctness for the computation of $n!$. In the second case, `OPER` is instantiated by the addition instruction `AddTo(dl.2,dl.1)`, leading to a proof of correctness for the computation of the n -th triangular number.

These case studies demonstrate that the developed framework provides a reusable methodology for the formal verification of a family of iterative algorithms sharing the same control-flow structure while differing only in the operation performed within the loop.

1. PRELIMINARIES

From now on $x_1, x_2, x_3, x_4, x_5, x_6$ denote objects, f denotes a function, j, k, m, n denote natural numbers, and i, i_1, i_2, i_3 denote integers.

Let a, b, c, x, y, z be objects. Let us observe that $(a, b, c) \mapsto (x, y, z)$ is finite.

Now we state the propositions:

- (1) $\text{dom}\langle x_1, x_2, x_3, x_4 \rangle = \{0, 1, 2, 3\}$.
- (2) $\text{len}\langle x_1, x_2, x_3, x_4 \rangle = 4$.
- (3) Let us consider functions f, g , and objects w, x, y, z . Suppose $\text{dom } f = \text{dom } g$ and $f(w) = g(w)$ and $f(x) = g(x)$ and $f(y) = g(y)$ and $f(z) = g(z)$. Then $f \upharpoonright \{w, x, y, z\} = g \upharpoonright \{w, x, y, z\}$.

2. SEQUENCES

Let us consider x_1, x_2, x_3, x_4 , and x_5 . The functor $\langle x_1, x_2, x_3, x_4, x_5 \rangle$ yielding a set is defined by the term

(Def. 1) $\langle x_1, x_2, x_3, x_4 \rangle \frown \langle x_5 \rangle$.

Let us note that $\langle x_1, x_2, x_3, x_4, x_5 \rangle$ is function-like and relation-like and $\langle x_1, x_2, x_3, x_4, x_5 \rangle$ is finite and transfinite sequence-like.

Now we state the propositions:

- (4) $\text{dom}\langle x_1, x_2, x_3, x_4, x_5 \rangle = \{0, 1, 2, 3, 4\}$. The theorem is a consequence of (1).
- (5) $\text{len}\langle x_1, x_2, x_3, x_4, x_5 \rangle = 5$.

Let us consider x_1, x_2, x_3, x_4 , and x_5 . Note that $\langle x_1, x_2, x_3, x_4, x_5 \rangle(0)$ reduces to x_1 and $\langle x_1, x_2, x_3, x_4, x_5 \rangle(1)$ reduces to x_2 and $\langle x_1, x_2, x_3, x_4, x_5 \rangle(2)$ reduces to x_3 and $\langle x_1, x_2, x_3, x_4, x_5 \rangle(3)$ reduces to x_4 and $\langle x_1, x_2, x_3, x_4, x_5 \rangle(4)$ reduces to x_5 .

Let us consider x_6 . The functor $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle$ yielding a set is defined by the term

(Def. 2) $\langle x_1, x_2, x_3, x_4, x_5 \rangle \frown \langle x_6 \rangle$.

Note that $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle$ is function-like and relation-like and $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle$ is finite and transfinite sequence-like.

Now we state the propositions:

- (6) $\text{dom}\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle = \{0, 1, 2, 3, 4, 5\}$. The theorem is a consequence of (4).
- (7) $\text{len}\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle = 6$.

Let us consider x_1, x_2, x_3, x_4, x_5 , and x_6 . Observe that $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle(0)$ reduces to x_1 and $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle(1)$ reduces to x_2 and $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle(2)$ reduces to x_3 and $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle(3)$ reduces to x_4 and $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle(4)$ reduces to x_5 and $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle(5)$ reduces to x_6 .

Now we state the proposition:

- (8) If $\langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle \subseteq f$, then $f(0) = x_1$ and $f(1) = x_2$ and $f(2) = x_3$ and $f(3) = x_4$ and $f(4) = x_5$ and $f(5) = x_6$. The theorem is a consequence of (6).

3. GENERAL THEORY OF SCM

From now on s denotes a state of **SCM**, I denotes an instruction of **SCM**, P denotes a instruction sequence of **SCM**, and X denotes a finite 0-sequence of the instructions of **SCM**.

Now we state the proposition:

- (9) Let us consider a data-location d . Then $i \in (\text{the values of } \mathbf{SCM})(d)$.

Let l_1 be a data-location and a be an integer. Let us observe that $l_1 \mapsto a$ is (the values of **SCM**)-compatible.

Let l_1, l_2 be data-locations and a, b be integers. One can verify that $[l_1 \mapsto a, l_2 \mapsto b]$ is (the values of **SCM**)-compatible.

Let l_1, l_2, l_3 be data-locations and a, b, c be integers. One can check that $(l_1, l_2, l_3) \mapsto (a, b, c)$ is (the values of **SCM**)-compatible.

Let d be a data-location and i be an integer. Note that $d \mapsto i$ is data-only as a part state of **SCM**.

Let p, q be data-only finite partial states of **SCM**. Observe that $p + q$ is data-only.

Let l_1, l_2 be data-locations and a, b be integers. Let us note that $[l_1 \mapsto a, l_2 \mapsto b]$ is data-only as a finite partial state of **SCM**.

Let l_1, l_2, l_3 be data-locations and a, b, c be integers. One can check that $(l_1, l_2, l_3) \mapsto (a, b, c)$ is data-only as a finite partial state of **SCM**.

Let us consider n and I . We say that I is n -dl.constant if and only if

- (Def. 3) for every instruction sequence P of **SCM** and for every state s of **SCM** and for every natural number k such that $\text{CurInstr}(P, \text{Comput}(P, s, k)) = I$ holds $(\text{Comput}(P, s, k+1))(\mathbf{d}_n) = (\text{Comput}(P, s, k))(\mathbf{d}_n)$.

Now we state the propositions:

- (10) If $m \neq j$, then $\text{AddTo}(\mathbf{d}_m, \mathbf{d}_n)$ is j -dl.constant.

- (11) If $m \neq j$, then $\text{SubFrom}(\mathbf{d}_m, \mathbf{d}_n)$ is j -dl.constant.

- (12) If $m \neq j$, then $\text{MultBy}(\mathbf{d}_m, \mathbf{d}_n)$ is j -dl.constant.

Let us consider m . Let n be a non zero natural number. Observe that $\text{AddTo}(\mathbf{d}_{m+n}, \mathbf{d}_m)$ is m -dl.constant and $\text{SubFrom}(\mathbf{d}_{m+n}, \mathbf{d}_m)$ is m -dl.constant and $\text{MultBy}(\mathbf{d}_{m+n}, \mathbf{d}_m)$ is m -dl.constant and $\text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)$ is 0-dl.constant and $\text{SubFrom}(\mathbf{d}_2, \mathbf{d}_1)$ is 0-dl.constant and $\text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)$ is 0-dl.constant.

Let us consider n . Let us note that there exists an instruction of **SCM** which is sequential and n -dl.constant.

Now we state the propositions:

- (13) Let us consider a sequential instruction I of **SCM**. Suppose $\text{CurInstr}(P, \text{Comput}(P, I) \text{ and } \mathbf{IC}_{\text{Comput}(P,s,n)} \in \text{dom } P$. Then $\mathbf{IC}_{\text{Comput}(P,s,n+1)} = \mathbf{IC}_{\text{Comput}(P,s,n)} + 1$.
- (14) Let us consider an n -started state s consisting of $\langle i_1, i_2, i_3 \rangle$. Then
 - (i) $s(\mathbf{d}_0) = i_1$, and
 - (ii) $s(\mathbf{d}_1) = i_2$, and
 - (iii) $s(\mathbf{d}_2) = i_3$.

Let us consider X . Note that there exists a instruction sequence of **SCM** which is X -extending.

Now we state the proposition:

- (15) Let us consider a state s of **SCM**. Suppose $\text{Initialize}((\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (i_1, i_2, i_3)) \subseteq s$. Then s is a state consisting of $\langle i_1, i_2, i_3 \rangle$.

4. CORRECTNESS OF ALGORITHMS COMPUTING OPERATIONS ON ONE PREVIOUS ELEMENT

Let O be an instruction of **SCM**. The functor **ResultOfOne(O)** yielding a finite 0-sequence of the instructions of **SCM** is defined by the term

(Def. 4) $\langle \text{if } \mathbf{d}_1 > 0 \text{ goto } 2, \text{halts}_{\text{SCM}}, O, \text{SubFrom}(\mathbf{d}_1, \mathbf{d}_0), \text{if } \mathbf{d}_1 = 0 \text{ goto } 1, \text{SCM} - \text{goto } 2 \rangle$

Now we state the propositions:

- (16) Suppose $\text{ResultOfOne}(I) \subseteq P$. Suppose $\mathbf{IC}_{\text{Comput}(P,s,k)} = 0$. Then
 - (i) if $(\text{Comput}(P, s, k))(\mathbf{d}_1) > 0$, then $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 2$, and
 - (ii) if $(\text{Comput}(P, s, k))(\mathbf{d}_1) \leq 0$, then $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 1$, and
 - (iii) for every j , $(\text{Comput}(P, s, k+1))(\mathbf{d}_j) = (\text{Comput}(P, s, k))(\mathbf{d}_j)$.

PROOF: If $(\text{Comput}(P, s, k))(\mathbf{d}_1) \leq 0$, then $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 1$ by [4, (9)]. $\square$

- (17) If $\text{ResultOfOne}(I) \subseteq P$, then P halts at 1. The theorem is a consequence of (8).
- (18) If $\text{ResultOfOne}(I) \subseteq P$, then for every k and j such that $\mathbf{IC}_{\text{Comput}(P,s,k)} = 1$ holds $\text{Comput}(P, s, k+j) = \text{Comput}(P, s, k)$. The theorem is a consequence of (17).

Let us consider k . Now we state the propositions:

(19) Suppose $\text{ResultOfOne}(\text{AddTo}(\mathbf{d}_m, \mathbf{d}_n)) \subseteq P$. Then suppose $\mathbf{IC}_{\text{Comput}(P,s,k)} = 2$. Then

- (i) $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 3$, and
- (ii) $(\text{Comput}(P, s, k+1))(\mathbf{d}_m) = (\text{Comput}(P, s, k))(\mathbf{d}_m) + (\text{Comput}(P, s, k))(\mathbf{d}_n)$,
and
- (iii) for every j such that $j \neq m$ holds $(\text{Comput}(P, s, k+1))(\mathbf{d}_j) = (\text{Comput}(P, s, k))(\mathbf{d}_j)$.

The theorem is a consequence of (8).

(20) Suppose $\text{ResultOfOne}(\text{SubFrom}(\mathbf{d}_m, \mathbf{d}_n)) \subseteq P$. Then suppose $\mathbf{IC}_{\text{Comput}(P,s,k)} = 2$. Then

- (i) $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 3$, and
- (ii) $(\text{Comput}(P, s, k+1))(\mathbf{d}_m) = (\text{Comput}(P, s, k))(\mathbf{d}_m) - (\text{Comput}(P, s, k))(\mathbf{d}_n)$,
and
- (iii) for every j such that $j \neq m$ holds $(\text{Comput}(P, s, k+1))(\mathbf{d}_j) = (\text{Comput}(P, s, k))(\mathbf{d}_j)$.

The theorem is a consequence of (8).

(21) Suppose $\text{ResultOfOne}(\text{MultBy}(\mathbf{d}_m, \mathbf{d}_n)) \subseteq P$. Then suppose $\mathbf{IC}_{\text{Comput}(P,s,k)} = 2$. Then

- (i) $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 3$, and
- (ii) $(\text{Comput}(P, s, k+1))(\mathbf{d}_m) = (\text{Comput}(P, s, k))(\mathbf{d}_m) \cdot (\text{Comput}(P, s, k))(\mathbf{d}_n)$,
and
- (iii) for every j such that $j \neq m$ holds $(\text{Comput}(P, s, k+1))(\mathbf{d}_j) = (\text{Comput}(P, s, k))(\mathbf{d}_j)$.

The theorem is a consequence of (8).

(22) Suppose $\text{ResultOfOne}(I) \subseteq P$. Then suppose $\mathbf{IC}_{\text{Comput}(P,s,k)} = 3$. Then

- (i) $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 4$, and
- (ii) $(\text{Comput}(P, s, k+1))(\mathbf{d}_1) = (\text{Comput}(P, s, k))(\mathbf{d}_1) - (\text{Comput}(P, s, k))(\mathbf{d}_0)$,
and
- (iii) for every j such that $j \neq 1$ holds $(\text{Comput}(P, s, k+1))(\mathbf{d}_j) = (\text{Comput}(P, s, k))(\mathbf{d}_j)$.

The theorem is a consequence of (8).

(23) Suppose $\text{ResultOfOne}(I) \subseteq P$. Then suppose $\mathbf{IC}_{\text{Comput}(P,s,k)} = 4$. Then

- (i) if $(\text{Comput}(P, s, k))(\mathbf{d}_1) = 0$, then $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 1$, and
- (ii) if $(\text{Comput}(P, s, k))(\mathbf{d}_1) \neq 0$, then $\mathbf{IC}_{\text{Comput}(P,s,k+1)} = 5$, and
- (iii) for every j , $(\text{Comput}(P, s, k+1))(\mathbf{d}_j) = (\text{Comput}(P, s, k))(\mathbf{d}_j)$.

PROOF: If $(\text{Comput}(P, s, k))(\mathbf{d}_1) \neq 0$, then $\mathbf{IC}_{\text{Comput}(P, s, k+1)} = 5$ by [4, (8)]. $\square$

(24) Suppose $\text{ResultOfOne}(I) \subseteq P$. Then suppose $\mathbf{IC}_{\text{Comput}(P, s, k)} = 5$. Then

- (i) $\mathbf{IC}_{\text{Comput}(P, s, k+1)} = 2$, and
- (ii) for every j , $(\text{Comput}(P, s, k+1))(\mathbf{d}_j) = (\text{Comput}(P, s, k))(\mathbf{d}_j)$.

The theorem is a consequence of (8).

Let us consider I . One can check that $\text{ResultOfOne}(I)$ is (the instructions of **SCM**)-valued and $\text{ResultOfOne}(I)$ is non halt-free.

Now we state the propositions:

(25) Suppose I is sequential and $\text{ResultOfOne}(I) \subseteq P$ and $\mathbf{IC}_{\text{Comput}(P, s, 0)} = 0$. Then $\mathbf{IC}_{\text{Comput}(P, s, k)} \in \{0, 1, 2, 3, 4, 5\}$.

PROOF: Define $\mathcal{P}[\text{natural number}] \equiv \mathbf{IC}_{\text{Comput}(P, s, \$1)} \in \{0, 1, 2, 3, 4, 5\}$. For every j such that $\mathcal{P}[j]$ holds $\mathcal{P}[j+1]$ by (16), (18), (8), [?, (3), (28)]. For every j , $\mathcal{P}[j]$ from [1, Sch. 2]. $\square$

(26) Suppose I is sequential and 0-dl.constant and $\text{ResultOfOne}(I) \subseteq P$. Let us consider a 0-started state s consisting of $\langle i, i_1, i_2 \rangle$. Then $(\text{Comput}(P, s, k))(\mathbf{d}_0) = i$.

PROOF: Define $\mathcal{P}[\text{natural number}] \equiv (\text{Comput}(P, s, \$1))(a) = i$. $\mathcal{P}[0]$. For every k such that $\mathcal{P}[k]$ holds $\mathcal{P}[k+1]$ by (25), (16), (18), (22). For every k , $\mathcal{P}[k]$ from [1, Sch. 2]. $\square$

(27) Suppose I is sequential and 0-dl.constant. Let us consider a instruction sequence P of **SCM**. Suppose $\text{ResultOfOne}(I) \subseteq P$. Let us consider a 0-started state s consisting of $\langle i_1, 0, i_2 \rangle$. Then

- (i) $\mathbf{IC}_{\text{Comput}(P, s, 0)} = 0$, and
- (ii) for every natural number k such that $k \geq 1$ holds $\mathbf{IC}_{\text{Comput}(P, s, k)} = 1$, and
- (iii) for every natural number k , $(\text{Comput}(P, s, k))(\mathbf{d}_0) = i_1$ and $(\text{Comput}(P, s, k))(\mathbf{d}_1) = 0$ and $(\text{Comput}(P, s, k))(\mathbf{d}_2) = i_2$, and
- (iv) $\text{LifeSpan}(P, s) = 1$.

The theorem is a consequence of (14), (16), (18), (26), and (8).

Let us consider an integer A , a natural number N , a instruction sequence P of **SCM**, and a 0-started state s consisting of $\langle 1, N, A \rangle$. Now we state the propositions:

(28) Suppose $\text{ResultOfOne}(I) \subseteq P$ and I is sequential and 0-dl.constant. Then suppose $(\text{Comput}(P, s, 2))(\mathbf{d}_1) = (\text{Comput}(P, s, 1))(\mathbf{d}_1)$ and for every natural number k , $(\text{Comput}(P, s, 4 \cdot k + 6))(\mathbf{d}_1) = (\text{Comput}(P, s, 4 \cdot k + 5))(\mathbf{d}_1)$. Then

- (i) P is halting on s , and
- (ii) $\mathbf{IC}_{\text{Comput}(P,s,0)} = 0$, and
- (iii) $(\text{Comput}(P, s, 1))(\mathbf{d}_2) = A$, and
- (iv) for every natural number k such that $0 < k < N$ holds $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k)} = 5$, and
- (v) for every natural number k such that $k < N$ holds $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+1)} = 2$ and $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+2)} = 3$ and $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+3)} = 4$ and $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1) = N - k$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) = N - k$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_1) = N - k - 1$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(\mathbf{d}_1) = N - k - 1$ and $(\text{Comput}(P, s, 4 \cdot k))(\mathbf{d}_1) > 0$ and $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1) > 0$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) > 0$, and
- (vi) if $N \neq 0$, then $\text{LifeSpan}(P, s) = 4 \cdot N$ and for every natural numbers k, j such that $k \geq N$ holds $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+j)} = 1$ and $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_1) = 0$.

The theorem is a consequence of (8), (14), (16), (6), (13), (26), (22), (23), (24), and (18).

- (29) Suppose $(I = \text{AddTo}(\mathbf{d}_2, \mathbf{d}_1) \text{ or } I = \text{SubFrom}(\mathbf{d}_2, \mathbf{d}_1) \text{ or } I = \text{MultBy}(\mathbf{d}_2, \mathbf{d}_1))$ and $\text{ResultOfOne}(I) \subseteq P$. Then

- (i) P is halting on s , and
- (ii) $(\text{Comput}(P, s, 1))(\mathbf{d}_2) = A$, and
- (iii) for every natural number k such that $0 < k < N$ holds $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k)} = 5$, and
- (iv) for every natural number k such that $k < N$ holds $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+1)} = 2$ and $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+2)} = 3$ and $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+3)} = 4$ and $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1) = N - k$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) = N - k$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_1) = N - k - 1$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(\mathbf{d}_1) = N - k - 1$ and $(\text{Comput}(P, s, 4 \cdot k))(\mathbf{d}_1) > 0$ and $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1) > 0$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) > 0$, and
- (v) if $N \neq 0$, then $\text{LifeSpan}(P, s) = 4 \cdot N$ and for every natural numbers k, j such that $k \geq N$ holds $\mathbf{IC}_{\text{Comput}(P,s,4 \cdot k+j)} = 1$ and $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_1) = 0$.

The theorem is a consequence of (8), (14), (16), (19), (20), (21), (22), (23), (24), and (18).

Let us consider i_1, i_2, k , and n . Let s be a 0-started state consisting of $\langle n, i_1, i_2 \rangle$, I be a sequential, 0-dl.constant instruction of **SCM**, and P be a (ResultOfOne(I))-extending instruction sequence of **SCM**. Observe that $(\text{Comput}(P, s, k))(\mathbf{d}_0)$ is natural.

Now we state the proposition:

(30) Suppose $I = \text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)$ or $I = \text{SubFrom}(\mathbf{d}_2, \mathbf{d}_1)$ or $I = \text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)$.

Let us consider a 0-started state s consisting of $\langle 1, n, i \rangle$, and a (ResultOfOne(I))-extending instruction sequence P of **SCM**. Then $(\text{Comput}(P, s, k))(\mathbf{d}_1)$ is natural. The theorem is a consequence of (27), (14), and (29).

Let us consider i, k , and n . Let s be a 0-started state consisting of $\langle 1, n, i \rangle$ and P be a (ResultOfOne($\text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)$))-extending instruction sequence of **SCM**. Let us observe that $(\text{Comput}(P, s, k))(\mathbf{d}_1)$ is natural.

Let P be a (ResultOfOne($\text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)$))-extending instruction sequence of **SCM**. Note that $(\text{Comput}(P, s, k))(\mathbf{d}_1)$ is natural.

The scheme *AddOfOneSch* deals with an integer $\mathcal{A}$ and a natural number $\mathcal{N}$ and a unary functor $\mathcal{F}$ yielding an integer and states that

(Sch. 1) For every (ResultOfOne($\text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)$))-extending instruction sequence P of **SCM** for every 0-started state s consisting of $\langle 1, \mathcal{N}, \mathcal{A} \rangle$, for every natural number k such that $k < \mathcal{N}$ holds $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_2) = \mathcal{A} + (\mathcal{F}(\mathcal{N}) - \mathcal{F}((\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1)))$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2) = \mathcal{A} + (\mathcal{F}(\mathcal{N}) - \mathcal{F}((\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) - 1))$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2)$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2)$ and for every natural numbers k, j such that $k \geq \mathcal{N}$ holds $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_2) = \mathcal{F}(\mathcal{N})$ and $(\text{Result}(P, s))(\mathbf{d}_2) = \mathcal{F}(\mathcal{N})$

provided

- $\mathcal{F}(0) = \mathcal{A}$ and
- for every natural number n , $\mathcal{F}(n + 1) = n + 1 + \mathcal{F}(n)$.

The scheme *MultOfOneSch* deals with an integer $\mathcal{A}$ and a natural number $\mathcal{N}$ and a unary functor $\mathcal{F}$ yielding an integer and states that

(Sch. 2) For every (ResultOfOne($\text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)$))-extending instruction sequence P of **SCM** for every 0-started state s consisting of $\langle 1, \mathcal{N}, \mathcal{A} \rangle$, for every natural number k such that $k < \mathcal{N}$ holds $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_2) = \mathcal{A} \cdot \frac{\mathcal{F}(\mathcal{N})}{\mathcal{F}((\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1))}$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2) = \mathcal{A} \cdot \frac{\mathcal{F}(\mathcal{N})}{\mathcal{F}((\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) - 1)}$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2)$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2)$ and for every natural numbers k, j such that $k \geq \mathcal{N}$ holds $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_2) = \mathcal{F}(\mathcal{N})$ and $(\text{Result}(P, s))(\mathbf{d}_2) = \mathcal{F}(\mathcal{N})$

provided

- $\mathcal{F}(0) = \mathcal{A}$ and
- $\mathcal{A} \neq 0$ and
- for every natural number n , $\mathcal{F}(n+1) = (n+1) \cdot \mathcal{F}(n)$.

Now we state the propositions:

- (31) Let us consider a $(\text{ResultOfOne}(\text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)))$ -extending instruction sequence P of **SCM**, and a 0-started state s consisting of $\langle 1, n, 0 \rangle$. Then
- (i) for every natural number k such that $k < n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_2) = \text{Triangle } n - \text{Triangle}(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1)$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2) = \text{Triangle } n - \text{Triangle}((\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) -' 1)$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2)$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2)$, and
 - (ii) for every natural numbers k, j such that $k \geq n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_2) = \text{Triangle } n$, and
 - (iii) $(\text{Result}(P, s))(\mathbf{d}_2) = \text{Triangle } n$.

PROOF: For every $(\text{ResultOfOne}(\text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)))$ -extending instruction sequence P of **SCM** and for every 0-started state s consisting of $\langle 1, n, 0 \rangle$, for every natural number k such that $k < n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_2) = 0 + (\mathcal{T}(n) - \mathcal{T}((\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1)))$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2) = 0 + (\mathcal{T}(n) - \mathcal{T}((\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) -' 1))$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2)$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2)$ and for every natural numbers k, j such that $k \geq n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_2) = \mathcal{T}(n)$ and $(\text{Result}(P, s))(\mathbf{d}_2) = \mathcal{T}(n)$ from *AddOfOneSch.* $\square$

- (32) Let us consider a $(\text{ResultOfOne}(\text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)))$ -extending instruction sequence P of **SCM**, and a 0-started state s consisting of $\langle 1, n, 1 \rangle$. Then
- (i) for every natural number k such that $k < n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_2) = \frac{n!}{(\text{Comput}(P, s, 4 \cdot k + 1))(\mathbf{d}_1)!}$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2) = \frac{n!}{((\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_1) -' 1)!}$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 2))(\mathbf{d}_2)$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(\mathbf{d}_2) = (\text{Comput}(P, s, 4 \cdot k + 3))(\mathbf{d}_2)$, and
 - (ii) for every natural numbers k, j such that $k \geq n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_2) = n!$, and
 - (iii) $(\text{Result}(P, s))(\mathbf{d}_2) = n!$.

PROOF: For every $(\text{ResultOfOne}(\text{MultBy}(c, b)))$ -extending instruction sequence P of **SCM** and for every 0-started state s consisting of $\langle 1, n, 1 \rangle$, for every natural number k such that $k < n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1))(c) = 1 \cdot \frac{\mathcal{F}(n)}{\mathcal{F}((\text{Comput}(P, s, 4 \cdot k + 1))(b))}$ and $(\text{Comput}(P, s, 4 \cdot k + 2))(c) = 1 \cdot \frac{\mathcal{F}(n)}{\mathcal{F}((\text{Comput}(P, s, 4 \cdot k + 2))(b) - 1)}$ and $(\text{Comput}(P, s, 4 \cdot k + 3))(c) = (\text{Comput}(P, s, 4 \cdot k + 2))(c)$ and $(\text{Comput}(P, s, 4 \cdot k + 4))(c) = (\text{Comput}(P, s, 4 \cdot k + 3))(c)$ and for every natural numbers k, j such that $k \geq n$ holds $(\text{Comput}(P, s, 4 \cdot k + 1 + j))(\mathbf{d}_2) = \mathcal{F}(n)$ and $(\text{Result}(P, s))(c) = \mathcal{F}(n)$ from *MultOfOneSch*. $\square$

5. EXAMPLES

The scheme *PartFuncFinPartStEx* deals with integers i_1, i_2 and a unary functor $\mathcal{F}$ yielding an integer and states that

- (Sch. 3) There exists a partial function f from $\text{FinPartSt}(\mathbf{SCM})$ to $\text{FinPartSt}(\mathbf{SCM})$ such that for every finite partial states p, q of **SCM**, $\langle p, q \rangle \in f$ iff there exists a natural number n such that $p = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (i_1, n, i_2)$ and $q = \mathbf{d}_2 \mapsto \mathcal{F}(n)$.

The scheme *PartFuncFinPartStUn* deals with integers i_1, i_2 and a unary functor $\mathcal{F}$ yielding an integer and states that

- (Sch. 4) For every partial functions f_1, f_2 from $\text{FinPartSt}(\mathbf{SCM})$ to $\text{FinPartSt}(\mathbf{SCM})$ such that for every finite partial states p, q of **SCM**, $\langle p, q \rangle \in f_1$ iff there exists a natural number n such that $p = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (i_1, n, i_2)$ and $q = \mathbf{d}_2 \mapsto \mathcal{F}(n)$ and for every finite partial states p, q of **SCM**, $\langle p, q \rangle \in f_2$ iff there exists a natural number n such that $p = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (i_1, n, i_2)$ and $q = \mathbf{d}_2 \mapsto \mathcal{F}(n)$ holds $f_1 = f_2$.

The functor **powerSCMFunction** yielding a partial function from $\text{FinPartSt}(\mathbf{SCM})$ to $\text{FinPartSt}(\mathbf{SCM})$ is defined by

- (Def. 5) for every finite partial states p, q of **SCM**, $\langle p, q \rangle \in it$ iff there exists a natural number n such that $p = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, 1)$ and $q = \mathbf{d}_2 \mapsto n!$.

Now we state the propositions:

- (33) Let us consider an object p . Then $p \in \text{dom}(\text{powerSCMFunction})$ if and only if there exists a natural number n such that $p = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, 1)$.
- (34) $(\text{powerSCMFunction})((\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, 1)) = \mathbf{d}_2 \mapsto n!$.

The scheme *AutonomicSch* deals with a partial function f from $\text{FinPartSt}(\mathbf{SCM})$ to $\text{FinPartSt}(\mathbf{SCM})$ and an instruction $\mathcal{I}$ of **SCM** and a unary functor $\mathcal{F}$ yielding a natural number and states that

(Sch. 5) For every finite partial state x of **SCM** such that $x \in \text{dom } f$ holds $\text{Initialize}(x)$ is $(\text{ResultOfOne}(\mathcal{I}))$ -autonomic provided

- $\mathcal{I} = \text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)$ or $\mathcal{I} = \text{SubFrom}(\mathbf{d}_2, \mathbf{d}_1)$ or $\mathcal{I} = \text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)$ and
- for every object x such that $x \in \text{dom } f$ there exists a natural number n such that $x = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, \mathcal{F}(n))$ and
- for every instruction sequences P, Q of **SCM** such that $\text{ResultOfOne}(\mathcal{I}) \subseteq P$ and $\text{ResultOfOne}(\mathcal{I}) \subseteq Q$ for every finite partial state x of **SCM** such that $x \in \text{dom } f$ for every states s_1, s_2 of **SCM** such that $\text{Initialize}(x) \subseteq s_1$ and $\text{Initialize}(x) \subseteq s_2$ for every natural number k , $(\text{Comput}(P, s_1, k))(\mathbf{d}_2) = (\text{Comput}(Q, s_2, k))(\mathbf{d}_2)$.

The scheme *HaltedSch* deals with a partial function f from $\text{FinPartSt}(\mathbf{SCM})$ to $\text{FinPartSt}(\mathbf{SCM})$ and an instruction $\mathcal{I}$ of **SCM** and a unary functor $\mathcal{F}$ yielding a natural number and states that

(Sch. 6) For every finite partial state x of **SCM** such that $x \in \text{dom } f$ holds $\text{Initialize}(x)$ is $(\text{ResultOfOne}(\mathcal{I}))$ -halted provided

- $\mathcal{I} = \text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)$ or $\mathcal{I} = \text{SubFrom}(\mathbf{d}_2, \mathbf{d}_1)$ or $\mathcal{I} = \text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)$ and
- for every object x such that $x \in \text{dom } f$ there exists a natural number n such that $x = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, \mathcal{F}(n))$.

The scheme *AutonomySch* deals with a partial function f from $\text{FinPartSt}(\mathbf{SCM})$ to $\text{FinPartSt}(\mathbf{SCM})$ and an instruction $\mathcal{I}$ of **SCM** and a unary functor $\mathcal{F}$ yielding a natural number and states that

(Sch. 7) For every finite partial state x of **SCM** such that $x \in \text{dom } f$ holds $\text{Initialize}(x)$ is an autonomy of $\text{ResultOfOne}(\mathcal{I})$ provided

- $\mathcal{I} = \text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)$ or $\mathcal{I} = \text{SubFrom}(\mathbf{d}_2, \mathbf{d}_1)$ or $\mathcal{I} = \text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)$ and
- for every object x such that $x \in \text{dom } f$ there exists a natural number n such that $x = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, \mathcal{F}(n))$ and
- for every instruction sequences P, Q of **SCM** such that $\text{ResultOfOne}(\mathcal{I}) \subseteq P$ and $\text{ResultOfOne}(\mathcal{I}) \subseteq Q$ for every finite partial state x of **SCM** such that $x \in \text{dom } f$ for every states s_1, s_2 of **SCM** such that $\text{Initialize}(x) \subseteq s_1$ and $\text{Initialize}(x) \subseteq s_2$ for every natural number k , $(\text{Comput}(P, s_1, k))(\mathbf{d}_2) = (\text{Comput}(Q, s_2, k))(\mathbf{d}_2)$.

Now we state the propositions:

- (35) Let us consider a finite partial state x of **SCM**. Suppose $x \in \text{dom}(\text{powerSCMFunction})$. Then $\text{Initialize}(x)$ is a autonomy of $\text{ResultOfOne}(\text{MultBy}(\mathbf{d}_2, \mathbf{d}_1))$.

PROOF: Set $I = \text{MultBy}(c, b)$. Define $\mathcal{F}(\text{natural number}) = 1$. For every object x such that $x \in \text{dom } pf$ there exists a natural number n such that $x = (a, b, c) \mapsto (1, n, \mathcal{F}(n))$. For every instruction sequences P, Q of **SCM** such that $\text{ResultOfOne}(I) \subseteq P$ and $\text{ResultOfOne}(I) \subseteq Q$ for every finite partial state x of **SCM** such that $x \in \text{dom } pf$ for every states s_1, s_2 of **SCM** such that $\text{Initialize}(x) \subseteq s_1$ and $\text{Initialize}(x) \subseteq s_2$ for every natural number k , $(\text{Comput}(P, s_1, k))(c) = (\text{Comput}(Q, s_2, k))(c)$ by (33), (15), [?, (17)], (14). For every finite partial state x of **SCM** such that $x \in \text{dom } pf$ holds $\text{Initialize}(x)$ is a autonomy of $\text{ResultOfOne}(I)$ from *AutonomySch*. $\square$

- (36) $\text{ResultOfOne}(\text{MultBy}(\mathbf{d}_2, \mathbf{d}_1)), \text{StartAt}(0, \mathbf{SCM})$ computes $\text{StartAt}(0, \mathbf{SCM})$. The theorem is a consequence of (33), (35), (34), (15), and (32).

The functor **sumSCMFunction** yielding a partial function from $\text{FinPartSt}(\mathbf{SCM})$ to $\text{FinPartSt}(\mathbf{SCM})$ is defined by

- (Def. 6) for every finite partial states p, q of **SCM**, $\langle p, q \rangle \in it$ iff there exists a natural number n such that $p = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, 0)$ and $q = \mathbf{d}_2 \mapsto \text{Triangle } n$.

Now we state the propositions:

- (37) Let us consider an object p . Then $p \in \text{dom}(\text{sumSCMFunction})$ if and only if there exists a natural number n such that $p = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, 0)$.

- (38) $(\text{sumSCMFunction})((\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2) \mapsto (1, n, 0)) = \mathbf{d}_2 \mapsto \text{Triangle } n$.

- (39) Let us consider a finite partial state x of **SCM**. Suppose $x \in \text{dom}(\text{sumSCMFunction})$. Then $\text{Initialize}(x)$ is a autonomy of $\text{ResultOfOne}(\text{AddTo}(\mathbf{d}_2, \mathbf{d}_1))$.

PROOF: Set $I = \text{AddTo}(c, b)$. Define $\mathcal{F}(\text{natural number}) = 0$. For every object x such that $x \in \text{dom } pf$ there exists a natural number n such that $x = (a, b, c) \mapsto (1, n, \mathcal{F}(n))$. For every instruction sequences P, Q of **SCM** such that $\text{ResultOfOne}(I) \subseteq P$ and $\text{ResultOfOne}(I) \subseteq Q$ for every finite partial state x of **SCM** such that $x \in \text{dom } pf$ for every states s_1, s_2 of **SCM** such that $\text{Initialize}(x) \subseteq s_1$ and $\text{Initialize}(x) \subseteq s_2$ for every natural number k , $(\text{Comput}(P, s_1, k))(c) = (\text{Comput}(Q, s_2, k))(c)$ by (37), (15), [?, (17)], (14). For every finite partial state x of **SCM** such that $x \in \text{dom } pf$ holds $\text{Initialize}(x)$ is a autonomy of $\text{ResultOfOne}(I)$ from *AutonomySch*. $\square$

- (40) $\text{ResultOfOne}(\text{AddTo}(\mathbf{d}_2, \mathbf{d}_1)), \text{StartAt}(0, \mathbf{SCM})$ computes $\text{StartAt}(0, \mathbf{SCM})$. The theorem is a consequence of (37), (39), (38), (15), and (31).

REFERENCES

- [1] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(1):41–46, 1990.
- [2] Grzegorz Bancerek, Czesław Byliński, Adam Grabowski, Artur Korniłowicz, Roman Matuszewski, Adam Naumowicz, Karol Pąk, and Josef Urban. Mizar: State-of-the-art and beyond. In Manfred Kerber, Jacques Carette, Cezary Kaliszyk, Florian Rabe, and Volker Sorge, editors, *Intelligent Computer Mathematics*, volume 9150 of *Lecture Notes in Computer Science*, pages 261–279. Springer International Publishing, 2015. ISBN 978-3-319-20614-1. doi:10.1007/978-3-319-20615-8_17.
- [3] Grzegorz Bancerek, Czesław Byliński, Adam Grabowski, Artur Korniłowicz, Roman Matuszewski, Adam Naumowicz, and Karol Pąk. The role of the Mizar Mathematical Library for interactive proof development in Mizar. *Journal of Automated Reasoning*, 61(1):9–32, 2018. doi:10.1007/s10817-017-9440-6.
- [4] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Formalized Mathematics*, 4(1):51–56, 1993.

Received July 20, 2026
