Encoding Finite Sequences in RCA_0

Duarte Maia

December 17, 2022

Contents

1 Introduction

This essay is intended as a long-form version of pages 66-68 of Simpson's book [1], in which he describes a way to encode finite sequences of natural numbers in RCA_0. It was borne of dissatisfaction: while very efficient in wordcount, Simpson's exposition came across (to me) as unmotivated and mysterious. In this essay, I seek to do the opposite, and write a relatively detailed account of how one may encode finite sequences in RCA_0.

The encoding below is by far not the simplest, and upon conclusion of this document I found a couple of evident ways in which the development could be made shorter and more elegant. However, strong as the temptation to simplify in order to make a sleeker proof, I am afraid that this would detract from the naturality of the approach, so I have opted to preserve it in its original state.

2 Prerequisites and Conventions

The reader is assumed to be familiar with basic notions of the system RCA_0, which in this context means pages 63-66 of Simpson [1], and some familiarity in working within this system. This means that we will use basic arithmetical facts and abuses of notation, not writing out proofs in more detail than is deemed necessary, with intuitive clarity being prioritized over formal completeness. The reader who is not yet comfortable working in this manner within RCA_0 may take it as an exercise to translate the proofs below into formal proofs in RCA_0.

Of the arithmetical facts that we will be assuming, the most notable is the notion of ordered pair; see Theorem II.2.2 in [1].

3 What We Mean by Encoding of Finite Sequences

3.1 Finite Sequences

In order to make the end goal clear, we begin by defining what is sought.

In RCA_0 it is very easy to encode what we mean by a function, and in particular a sequence: it is a set F of ordered pairs $\langle x, y\rangle$ such that for all x in some domain X there exists exactly one $y \in \mathbb{N}$ such that $\langle x, y\rangle \in F$. Thus, we have a way to define the notion of finite sequence: a finite sequence is a set F as above (i.e. a function) whose domain is $X = \{x \in \mathbb{N} \mid x < n\}$ for some fixed $n \in \mathbb{N}$ (called the *length of the sequence*).

Now, the problem with this 'encoding' is that it encodes sequences as second order objects, when we know *a posteriori* that finite sequences can be encoded as first order objects, which is very useful. Thus, what we seek is a correspondence between finite sequences and a certain subset of the natural numbers, which will eventually be called $\mathrm{Seq}(\mathbb{N}) \subseteq \mathbb{N}$. Moreover, from some $s \in \mathrm{Seq}(\mathbb{N})$ we wish to perform certain operations, such as

- The length of the sequence, n,
- (Uniformly) recovering the k-th element of the sequence, for $k < n$,
- Constructing the empty sequence,
- Appending an element to the end of the sequence,
- Concatenating sequences,
- Comparing sequences lexicographically, etc.

As it turns out, the first two operations are enough to do pretty much anything one would want to do with sequences; in section 6 we use them to construct the appending map and the factorial function, and in section II.3 of Simpson [1] sequences are used to prove that the universe of functions in RCA_0 is closed under primitive recursion and minimization; the operations above are more than enough for this purpose.

3.2 Finite Sets

An alternative approach to this problem (which Simpson takes) is to encode finite sets instead of finite sequences. Indeed, from a finite set one may encode a finite sequence, or really any function with finite domain, in much the same way that one encodes functions from sets.

On the other hand, from an encoding of finite sequences it is easy to encode finite sets. Indeed, given a set X bounded by some integer n, it may be represented by a sequence of length n, whose k-th element is zero or one depending on whether $k \in X$.

We will not take this approach here.

4 The Bounds of the Problem: Limits to our Approach

It is known *a posteriori* that the things we can do in RCA_0 correspond roughly to those which can be done effectively. As such, it would be expected that any effective encoding of sequences would suffice. However, this is not the case, as sequences are necessary to do effective tasks; more precisely, they are necessary to do effective tasks with a variable number of steps.

This places a strong restriction on our encoding. Besides needing to be effective, we need that the extraction of the entries of the sequence needs to be doable in a fixed number of steps. This is no easy feat. Let us look at some approaches which are invalidated by this requirement.

- Linked lists: This is a recursive way to encode arbitrary length lists using ordered pairs. It works as follows. First, fix a symbol ε which is not an ordered pair (this can be done using a non-surjective encoding for pairs). Then, by fiat, we declare ε to be the empty list. Then, a list with one element x would be the pair $\langle x, \varepsilon \rangle$. A list with the elements x, y would correspond to $\langle x, \langle y, \varepsilon \rangle \rangle$, and so on.

  Despite being very elementary, this approach does not work in RCA_0 because recovering the n-th element of a list takes (heuristically) n operations, and so cannot be performed before we have a way to chain variably many operations.

- Binary: One could attempt to encode a finite set as the collection of nonzero digits of a certain number n, when n is written in binary. Recovering the k-th binary digit of n is not a terrible affair: first we define n_0 as the remainder of the division of n by 2^{k+1}, and then we check whether $n_0 \geq 2^k$. Both of these steps are easily done using bounded quantifiers, with the exception of powers. Indeed, taking powers is not built into RCA_0 (though it is built into other systems such as EFA, in which this approach would work), and while powers may be computed in RCA_0, their computation requires a variable number of steps, and so cannot easily be done until we have a notion of sequence.

- Prime decomposition: This is perhaps the most obvious encoding for a mathematician. Given a sequence $x_0, \ldots, x_n$ encode it as the number $p_0^{x_0} \ldots p_n^{x_n}$, where $p_0, p_1, \ldots$ is an enumeration of the prime numbers. Unfortunately, not only does extraction of the sequence entries also require powers, it moreover requires an enumeration of the prime numbers within the language.

5 Our Approach: Encoding Sequences

So, given that we are now more knowledgeable about what we *can't* do, let's talk about what we can do. Evidently we can add and multiply, but this will not give us much expressive power. The operations we have access to for the time being which give us the most expressive power are elementary number theory operations, such as:

- We can do Euclidean division, recovering both the quotient and the remainder,
- We can check divisibility of a number by another,
- We can find the minimal element of a nonempty set.

Euclidean division in particular provides us with a way to encode a list of numbers into a single natural. The idea is as follows: given a 'seed number' N, we take the remainder of the division of N by several distinct numbers $a_0, a_1, \ldots, a_{n-1}$ to obtain the encoded sequence $x_0, x_1, \ldots, x_{n-1}$. In the following, we use the notation $N \bmod a$ for the remainder of the division of N by a.

It is not obvious what the numbers $a_0, \ldots, a_{n-1}$ should be, though we certainly want them to be easily expressible, so it is reasonable to consider an arithmetical sequence $a_k = a_0 + kb$. However, in order to ensure that $N \bmod a_k$ can be arranged to be whatever we want, we will need to make sure that the elements of a_k don't have any interdependencies, in some sense. As an example of something that could go wrong, suppose that $b = 1$. Then, if a_0 is even then a_2 is also even, and this implies that $N \bmod a_0$ and $N \bmod a_2$ both have the same parity (which is the same as the parity of N). If a_0 were odd instead, the same argument holds for a_1 and a_3.

The problem that arises when $b = 1$ is that some elements of the sequence have common divisors. This problem also holds if a_0 is poorly chosen; for example, if a_0 is even. Thus, the first step to find our encoding is to show that this can be avoided. Note that we must also show that a_0 may be made arbitrarily large, as $(N \bmod a_k) < a_k$ always.

Theorem 1. RCA_0 proves the following. Given $n \in \mathbb{N}$ and $m \in \mathbb{N}$, there exist $a_0, b \in \mathbb{N}$ such that $m < a_0$, and $a_0 + k_0 b$ and $a_0 + k_1 b$ have no common divisors (except for 1) for all distinct $k_0, k_1 < n$.

Proof Sketch: Before performing the actual proof, we give a plausibility argument by reasoning 'in usual mathematics'.

First, let us consider for simplicity the case $a_0 = 1$. The requirement that $1 + k_0 b$ be coprime with $1 + k_1 b$ for $k_0 < k_1 < n$ is equivalent to requiring that $1 + k_0 b$ be coprime with $(k_1 - k_0)b$. Now, we claim that $b = n!$ works. Indeed, in this case, all prime divisors of the right-hand side are less than n, while on the left-hand side we have a number whose remainder of division by all such primes is one.

Now it should be clear that we can moreover choose arbitrarily large values of a_0, as adding any multiple of $n!$ to a_0 will preserve this property.

Now, let us investigate how we could implement this in RCA_0. Unfortunately, the factorial operation is not one we have access to yet, because it is computed by recursion and this requires sequences. However, with Σ^0_1 induction we can prove the following:

$$\mathrm{RCA}_0 \vdash \begin{array}{l}\text{For all } n \in \mathbb{N}, \text{ there exists } r \in \mathbb{N},\ r \neq 0 \\ \text{which is divisible by all } k < n.\end{array} \tag{1}$$

Unfortunately, (1) is not quite enough for the proof idea above to work. This is because we have no guarantee that r is 'not too big', in the sense that it has no prime divisors greater than n. Thus, we want (and will prove) a slightly stronger existence theorem (though the proof is mostly the same):

$$\mathrm{RCA}_0 \vdash \begin{array}{l}\text{For all } n \in \mathbb{N}, \text{ there exists } r \in \mathbb{N},\ r \neq 0 \text{ which is} \\ \text{divisible by all } k < n, \\ \text{and whose prime divisors are all less than } n.\end{array} \tag{2}$$

Now we consider b equal to this value of r, and a_0 equal to $mr+1$. We claim that the only common divisor of $a_0 + k_0 b$ and $\Delta k\, b$ for $k_0, \Delta k < n$, $\Delta k \neq 0$, is 1. To do so, consider the set

$$X = \{\, d \in \mathbb{N} \mid d \neq 1 \text{ and } d \text{ divides } a_0 + k_0 b \text{ and } \Delta k\, b \,\}. \tag{3}$$

Note that this set can be constructed in RCA_0 because it is made by comprehension over a bounded quantifier formula. Now, we claim that X is empty, and to prove it, we suppose that it is not, and let d_0 be some element of X.

We have no guarantee that d_0 is prime, so we make it so. Indeed, it can be proven in RCA_0 that every $d_0 \neq 1$ has a prime divisor p, so we pick such a prime. Note that $p \neq 1$ and $p \mid d$, so p is also in X.

Now, we know that $p \mid \Delta k\, b$, so therefore $p \mid \Delta k$ or $p \mid b$. If $p \mid \Delta k$, then $p \leq \Delta k$ and hence $p < n$. On the other hand, if $p \mid b$, by construction of b (2) we also have $p < n$.

Now, on the other hand, we show that p does not divide $a_0 + k_0 b = 1 + (m + k_0)r$. To this effect, consider the remainder of the division of this expression by p. Indeed, by (2) we have that r is of the form pq for some q, and thus we can easily show that

$$a_0 + k_0 b = [(m + k_0)q]p + 1, \tag{4}$$

hence in particular the remainder of Euclidean division is 1, and hence $p \nmid a_0 + k_0 b$. This contradicts the hypothesis that $p \in X$, and since a contradiction was sought, we have finally shown that X is empty. In other words, $a_0 + k_0 b$ and $\Delta k\, b$ are coprime.

To conclude the proof of theorem 1 in RCA_0, it is a simple exercise to show that x and y with $x < y$ are coprime iff x and $y - x$ are coprime, and then apply this exercise to prove that all elements of $\{a_0 + kb \mid k < n\}$ are coprime. □

Now that we have built the desired sequence (a_k) by which we will take the remainders, we must build the large value of N such that $N \bmod a_k = x_k$, where $x_0, \ldots, x_{n-1}$ is the sequence to be encoded.

The statement that such a value of N exists is easily seen to be equivalent to a particular case of the famous Chinese Remainder Theorem, so now it remains to prove it in RCA_0.

Theorem 2. Let $(x_k)_{k<n}$ be a finite sequence of natural numbers[1]. Let $a_k = a_0 + kb$ with a_0 and b as in theorem 1, with m an upper bound for the sequence.[2] Then, there exists $N \in \mathbb{N}$ such that

$$x_k = N \bmod a_k, \text{ for } k < n. \tag{5}$$

Proof Sketch: First, let us look at the proof of this fact in 'ordinary mathematics'. The basic idea is to look at some N of the form

$$N = y_0 a_1 \ldots a_{n-1} + a_0 y_1 a_2 \ldots a_{n-1} + \cdots + a_0 \ldots a_{n-2} y_{n-1}. \tag{6}$$

For such N, it is easy to see that when taking the remainder modulo a_k almost all the terms vanish, so that

$$N \bmod a_k = (y_k a_0 \ldots a_{k-1} a_{k+1} \ldots a_{n-1}) \bmod a_k \tag{7}$$

Thus, it suffices to find y_k for each k such that (7) holds. Now we recall the following facts from elementary number theory:

a) If P is the product of integers which are coprime with a_k, then P itself is coprime with a_k, and

b) If P is coprime with a_k it has an inverse modulo a_k. In other words, there exists Q such that $PQ \bmod a_k = 1$.

The first statement is a trivial consequence of the definitions (and easy to prove in RCA_0), and the second statement follows from the Euclidean algorithm (and hence less easy to prove in RCA_0).

Unfortunately, expressions with ellipses such as (6) are not allowed in RCA_0, so to implement this proof in RCA_0 we will need to change our approach a little bit.

[1] Seen as a function from $\{k \in \mathbb{N} \mid k < n\}$ to $\mathbb{N}$.

[2] It is easy to prove by Σ^0_1 induction in n that this does exist in RCA_0.

In our approach, we will begin by constructing numbers that take the place of the summands in (6). As a first lemma, we 'construct $a_0 \dots \widehat{a_k} \dots a_{n-1}$'.

$$\mathrm{RCA}_0 \vdash \begin{array}{l} \text{For each } k < n, \text{ there exists } s \in \mathbb{N} \text{ such that} \\ s \bmod a_\ell = 0 \text{ for } \ell < n,\ \ell \neq k, \text{ and } s \text{ is coprime} \\ \text{with } a_k. \end{array} \tag{8}$$

The proof of (8) is messy but can be done by Σ^0_1 induction by proving the following statement. Let k, a_0, and b be fixed natural numbers:

$$\mathrm{RCA}_0 \vdash \begin{array}{l} \text{For each } n \in \mathbb{N}, \text{ there exists } N \in \mathbb{N} \text{ such that, if all distinct pairs} \\ \text{of elements of } \{a_0 + \ell b\}_{\ell < n \text{ or } \ell = k} \text{ are coprime, } N \text{ is coprime with} \\ a_0 + kb, \text{ and } N \text{ is divisible by } a_0 + \ell b \text{ for all } \ell < n,\ \ell \neq k. \end{array} \tag{9}$$

The base case $n = 0$ is done by setting $N = 1$. The induction step is done by multiplying the N obtained from the induction hypothesis by $a_0 + nb$ if $n \neq k$, and keeping the same value of N if $n = k$. This requires the use of the lemma: 'if x and y are coprime with a_n then so is xy', which we leave to the reader to verify is true in RCA_0.

If we fix n and apply (9) to a_0, b given by theorem 1, we have a proof of (8).

Now, we wish to find the equivalent to the y_k in (6). Classically, this is equivalent to the theorem that if x is coprime with n then it is invertible modulo n. We phrase it as such:

$$\mathrm{RCA}_0 \vdash \begin{array}{l} \text{If } x \text{ is coprime with } n \text{ and } y < n \text{ then there exists} \\ z \in \mathbb{N} \text{ such that } xz \bmod n = y. \end{array} \tag{10}$$

A classical proof in ordinary mathematics goes as follows. First, use Euclid's algorithm to prove (a particular case of) the Darboux theorem, which is that there exist integers z and q such that $xz - nq = 1$. Replacing (z, q) by $(z + kn, q+kn)$ for high enough k, we may assume that z and q are in fact nonnegative integers, and multiplying z and q by y, we may assume that $xz - nq = y$ instead. Finally, using the fact that $y < n$, we obtain that $xz \bmod n = y$ as desired.

A proof of this fact in RCA_0 still takes some work, so we postpone it to theorem 3 below.

We are now finally able to conclude the proof of theorem 2. We prove the following by Σ^0_1 induction on $\nu \in \mathbb{N}$, which morally takes the place of the index in the sum

$$N = \sum_{\nu=0}^{n-1} y_\nu \, a_0 \dots \widehat{a_\nu} \dots a_{n-1}. \tag{11}$$

Fixed a finite sequence $(x_k)_{k<n}$ and a_k as in theorem 1 (with m upper bound for (x_k)),

$$\mathrm{RCA}_0 \vdash \begin{array}{l} \text{For each } \nu \in \mathbb{N}, \text{ there exists } N \in \mathbb{N} \text{ such that,} \\ \text{if } \nu \leq n, \text{ for all } k < \nu \text{ we have } x_k = N \bmod a_k, \\ \text{and for } \nu \leq k < n \text{ we have } N \bmod a_k = 0. \end{array} \tag{12}$$

The base case $\nu = 0$ holds for $N = 0$, so it suffices to perform the induction step. Thus, we assume that the statement is true for some fixed ν, and prove it for $\nu + 1$. Assume that $\nu + 1 \leq n$ as otherwise the statement is also vacuously true.

Let N be previously built such that $x_k = N \bmod a_k$ for $k < \nu$ and $N \bmod a_k = 0$ for $\nu \leq k < n$. We wish to construct $N' \in \mathbb{N}$ such that $N' \bmod a_k = N \bmod a_k$ for $k < n$, $k \neq \nu$, and such that $N' \bmod a_\nu = x_\nu$. We construct N' by adding to N an appropriate number s. In particular, we construct s using (8), and by (10) we may replace s by an appropriate multiple of itself such that $s \bmod a_\nu = x_\nu$.

This concludes the proof of (12), and so, applying it to $\nu = n$, we finally complete the proof of theorem 2. □

Theorem 3. RCA_0 proves: If x is coprime with n and $y < n$ then there exists $z \in \mathbb{N}$ such that $xz \bmod n = y$.

Proof Sketch: As we have mentioned before, the usual proof of this fact is via the Darboux theorem, which in turn is usually shown using Euclid's algorithm. We do not yet have access to Euclid's algorithm (we need sequences to implement it), but this algorithm can actually be replaced by minimization. In particular, we may try instead find the minimal possible positive value of $xz - nq$ as z and q range over the positive integers. Now, this approach requires modification to work, as the set of all these values is not constructible in RCA_0, since it requires Σ^0_1 comprehension. However, we know *a posteriori* that z may be chosen less than n, and in this case (for positive $xz - nq$) q will be less than x. Thus, we may construct by bounded comprehension

$$X(x,n) = \{\, d \in \mathbb{N} \mid d \neq 0 \text{ and } \exists_{z<n} \exists_{q<x}\, xz = qn + d \,\} \tag{13}$$

and set $d(x,n)$ equal to the minimal element of $X(x,n)$. This exists because $X(x,n)$ is nonempty, as $x \in X(x,n)$ so long as $x > 0$ and $n > 1$, as $x \in X(x,n)$. Thus, *in the following we assume* $x \neq 0$ *and* $n > 1$. We leave it as an exercise to the reader to verify that (10) holds if $x = 0$ or $n \leq 1$.

We wish to show that $d_0 = d(x,n)$ is a common divisor of x and n. Suppose first that d_0 does not divide x. Then, we may consider $d_1 = x \bmod d_0$. It is clear that $0 < d_1 < d_0$, so if we prove that $d_1 \in X(x,n)$ we obtain a contradiction, from which we conclude that $d_0 \mid x$.

Thus, we write $d_0 = Qx + d_1$. Since $d_1 \in X(x,n)$ we may find z and q such that $xz = qn + d_0$, and putting these two equalities together we obtain

$$xz = qn + Qx + d_1. \tag{14}$$

Thus, we get $xz' = qn + d_1$ for some q and $z' = z - Q$ (Left to reader: verify that $z \geq Q$ and so z' is a well-defined natural). Now, we know that $q < x$ by hypothesis, but it remains to show that $z' < n$. To do so, note that $xz' = qn + d_1 \leq (x-1)n + d_1$. If we can show that $d_1 < n$ we have $xz' < xn$, hence, since $x \neq 0$, we get $z' < n$. To show that $d_1 < n$ we remark that $d_2 = x \bmod n \in X(x,n)$, as

$$x = qn + d_2, \tag{15}$$

with $qn \leq x$ and therefore, since $n > 1$, we have $q < x$. Moreover, $d_2 \neq 0$ as otherwise n would be a common divisor of x and n, which contradicts their coprimality (because $n > 1$). In conclusion, $d_1 < d_0 \leq d_2 < n$, and so by the previous paragraph we do indeed have $z' < n$.

Now we show that d_0 is a divisor of n. Similarly to before, consider $d_1 = d_0 \bmod n$, hence $d_0 = Qn + d_1$. Then, we have

$$xz = qn + Qn + d_1. \tag{16}$$

Thus, we conclude that $xz = q'n + d_1$ with $q' = q + Q$ and using a similar argument as before we conclude that $xn > q'n$ hence $x > q'$. Thus, $d_1 \in X(x, n)$ and we obtain a contradiction, hence $d_0 \mid n$.

We finally have a proof in RCA_0 of theorem 3, as if x is coprime with n then $d(x, n)$ may only be equal to one, hence there exist z and q such that $xz = qn+1$ and thus $xz \bmod n = 1$. It is thus easy to verify that for any $y < n$ we have $xzy \bmod n = y$. □

6 Operations on Sequences

Theorem 2 shows that any sequence can be encoded as a 4-uple $s = \langle n, N, a_0, b\rangle$ (called a *code for the sequence*), where n is the length of the sequence and N, a_0, b are as in theorem 2.

It is evident that the length of s can be obtained as the first entry of the 4-uple, and the k-th entry of the sequence is given by the remainder of division of N by $a_0 + kb$. Both of these are represented by bounded quantifier formulas.

In general, a sequence will have multiple distinct codes. This is undesirable, so for any given sequence we associate to it a canonical code: the minimal value of s which encodes it. We define $\mathrm{Seq}(\mathbb{N})$ as the set of all $s \in \mathbb{N}$ which are minimal codes for some sequence. This set exists by bounded quantifier comprehension, over the predicate: ‘for all $s' < s$, either the length of s' is different from the length of s, or there exists some k less than this length such that $s_k \neq s'_k$’. Thus, ‘the set of finite sequences of natural numbers’ is a well-defined set in RCA_0.

6.1 Application: Appending

Let $s \in \mathrm{Seq}(\mathbb{N})$, and $q \in \mathbb{N}$. Then, we may consider the sequence obtained by appending q to the end of s. We will show that this is well-defined in RCA_0, and in fact we can even construct the append function $A \colon \mathrm{Seq}(\mathbb{N}) \times \mathbb{N} \to \mathrm{Seq}(\mathbb{N})$ in RCA_0. The methods that we will use for this purpose can be adapted with very little modification to construct concatenation of sequences, restriction to a subsequence, among others.

First, we show that there exists a code for the sequence obtained by appending q to s. The proof has very little substance. Simply put, from $s = \langle n, N, a_0, b\rangle$ construct the sequence S which s is a code of, by setting

$$S = \{\, \langle x, y\rangle \mid x < n \text{ and } y = N \bmod (a_0 + xb)\,\}. \tag{17}$$

Then, append q to S by setting $S' = S \cup \{\langle n, q \rangle\}$. Finally, consider the code $s' \in \mathrm{Seq}(\mathbb{N})$ of S'. This proves that such a code exists.

Now, the reason we have provided the proof is because it does not look effective. Indeed, it requires passing through second order. As such, it would not appear at a glance that the append function could be defined in RCA_0. However, the fact that we have shown that this code exists and is unique suffices to build this function, because crucially, *we can verify if s' is the code of the sequence S' above without going through second order*. Indeed, it is a first order bounded quantifier predicate: check that $s' \in \mathrm{Seq}(\mathbb{N})$, that the length of s' is the length of s plus one, that $s'_k = s_k$ for k less than this length, and check that $s'_n = q$. As such, we may define the append function:

$$A = \left\{ \langle \langle s, q \rangle, s' \rangle \;\middle|\; \begin{array}{l} s, s' \in \mathrm{Seq}(\mathbb{N}) \text{ and } s' \text{ codifies the sequence} \\ \text{obtained by appending } q \text{ to } s \end{array} \right\}. \tag{18}$$

As we have mentioned before, these methods may be used with virtually no modification to implement several other common operations.

6.2 Application: Factorial Function

We now apply sequences as a tool to construct the factorial function in RCA_0. In other words, we will see that RCA_0 shows that there exists a unique function $F\colon \mathbb{N} \to \mathbb{N}$ such that $F(0) = 1$ and $F(n+1) = (n+1)F(n)$. Note that this example may easily be generalized to show that RCA_0 is closed under primitive recursion.

Uniqueness is easy to prove by induction. Given F and G satisfying this recurrence, define $X = \{n \in \mathbb{N} \mid F(n) = G(n)\}$, which exists by bounded quantifier comprehension. Then, the recurrence may easily be used to show by set induction that $X = \mathbb{N}$.

To show existence is where we apply the machinery of sequences. Define F as follows by Δ^0_1 comprehension:

$$F = \left\{ \langle x, y \rangle \;\middle|\; \begin{array}{l} \text{There exists a sequence } s \in \mathrm{Seq}(\mathbb{N}) \text{ of} \\ \text{length } x+1 \text{ such that } s_0 = 1, \text{ for all } k < x \\ \text{we have } s_{k+1} = (k+1)s_k, \text{ and } s_x = y. \end{array} \right\}, \tag{19}$$

This formula is Δ^0_1 because RCA_0 proves that such an s always exists, and that it is unique. Uniqueness is done similarly to uniqueness of F itself, and existence is done by Σ^0_1 induction, using the results from section 6.1. This shows that the 'exists' in φ may be replaced by a 'for all', which proves that φ is a Δ^0_1 predicate.

It remains to verify that F itself is a function and that it satisfies the recursion. By uniqueness and existence of s (with a given length), we conclude that for each x there exists exactly one y with $\langle x, y \rangle \in F$, hence F is a function. To verify that it satisfies the recursion, let s be the sequence of length $x+1$ satisfying the recurrence. Then, if s' is obtained by removing the last element,

s' also satisfies the recurrence. Hence,

$$F(x+1) = s_{x+1} = (x+1)s_x = (x+1)s'_x = (x+1)F(x). \tag{20}$$

The verification that $F(0) = 1$ is obvious, and thus F does satisfy the recurrence as desired. We have hence constructed the factorial function.

References

[1] Stephen G Simpson and Stephen George Simpson. *Subsystems of second order arithmetic*, volume 1. Cambridge University Press, 2009.