SI1001 Theory of Computation
Homework 1
Recursor for the Natural Numbers

Andrés Sicard-Ramírez

21st February 2025

1 Deadline

See the course homepage.

2 Recursor for Natural Numbers

Definition 2.1. The set of the natural numbers Nat can be inductively defined by

(i) $\mathrm{Z} \in \mathrm{Nat}$,

(ii) If $n \in \mathrm{Nat}$ then $\mathrm{S}\, n \in \mathrm{Nat}$.[1]

We can also use rule sets (inference rules) for inductively defined sets.

Definition 2.2. The set of the natural numbers Nat inductively defined by the following rule sets:

$$\frac{}{\mathrm{Z} \in \mathrm{Nat},} \qquad \frac{n \in \mathrm{Nat}}{\mathrm{S}\, n \in \mathrm{Nat}.}$$

Let A be a set. Usually a recursive function f on natural numbers has the shape

$$\begin{aligned} f &: \mathrm{Nat} \to A \\ f\, \mathrm{Z} \quad &= a, \\ f\, (\mathrm{S}\, n) &= h\, n\, (f\, n), \end{aligned} \tag{1}$$

[1] We wrote '$\mathrm{S}\, n$' instead of '$\mathrm{S}(n)$' following the notation for writing function application used in functional programming and lambda calculus.

where $a \in A$ and $h : \mathrm{Nat} \to \mathrm{A} \to \mathrm{A}$.[2] The function f applied to Z returns the constant value a. The function f applied to $\mathrm{S}\, n$ returns the value of the function h applied to both n and the recursive call $f\, n$.

Example 2.3. Using the pattern in (1) we define the addition of natural numbers by recursion on the second argument by

$$\begin{aligned} &\mathrm{add} : \mathrm{Nat} \to \mathrm{Nat} \to \mathrm{Nat} \\ &\mathrm{add}\, m\, \mathrm{Z} \quad\;\; = m, \\ &\mathrm{add}\, m\, (\mathrm{S}\, n) = \mathrm{S}\, (\mathrm{add}\, m\, n). \end{aligned}$$

Definition 2.4. The recursive pattern in (1) can be represented by a *recursor* for natural numbers defined by the following equations:

$$\begin{aligned} &\mathrm{rec} : A \to (\mathrm{Nat} \to A \to A) \to \mathrm{Nat} \to A, \\ &\mathrm{rec}\, a\, h\, \mathrm{Z} \quad\;\; = a, \\ &\mathrm{rec}\, a\, h\, (\mathrm{S}\, n) \; = h\, n\, (\mathrm{rec}\;\, a\, h\, n). \end{aligned}$$

The recursor rec is a higher-order function with three arguments: A first element of type A, a higher-order function $\mathrm{Nat} \to A \to A$ and a natural number Nat. For defining a recursive function like (1) using rec we following the approach:

(i) The value of $f\, \mathrm{Z}$ is the first argument to rec.

(ii) The function h in (1) must be the second argument of rec.

(iii) The third argument of rec is the argument used for the recursion in f.

Example 2.5. We define addition on natural numbers using rec from the add function in Example 2.3.

$$\begin{aligned} &\mathrm{addR} :: \mathrm{Nat} \to \mathrm{Nat} \to \mathrm{Nat} \\ &\mathrm{addR}\, m\, n = \mathrm{rec}\, n\, (\lambda x.\lambda y.\, \mathrm{S}\, y) m. \end{aligned}$$

Remark. Note that the function addR in the above example was defined only by *one* equation. This is a characteristic of any function defined by rec.

[2] The function $h : \mathrm{Nat} \to A \to A$ corresponds to the curryfication of a function $h' : \mathrm{Nat} \times A \to A$.

3 Assignment (95%)

(i) (20%) Implement in some programming language the recursor for natural numbers and four primitive recursive functions.[3]

(ii) (5%) To document (in English) your source code. The documentation should explain your solution.

(iii) (70%) Oral explanation of your solution.

4 Requirements (5%)

(i) The homework should be solved with other student taking the course.

(ii) To add to the repository a `README.md` file (Markdown format) in English containing the following information:

- Your(s) full name(s).
- Versions used of operating system, compiler and tools in your implementation.
- Detailed instructions for running your implementation.
- Any information (books, articles, videos, AIs, repositories, etc.) you did use for the homework.

(iii) Do not include unnecessary files or directories in the repository.

5 Clean code

Before submitting your code, which includes your `README.md` file, clean it up:

- Does not have long lines (at most 80 columns).
- Has an uniformly indentation (we recommended two characters).
- Has a consistent layout.
- Has good comments.
- Has no junk (unused code, commented code, unnecessary code).
- Has no overly complicated function definitions.
- Does not contain any repetitive code.

[3] I suggest to use a functional programming language because the recursor rec is a higher-order function.

- Has no tabs.
- Has no unnecessary spaces at the end of a lines, or empty lines at the end of a file.
- Has spell-checked comments.

6 Delivery

I shall send the GitHub Education link to the final project via EAFIT Interactiva.

7 From the coordination

El control de versiones no es solamente un herramienta que facilitará la comunicación entre los miembros del grupo y la administración de los cambios al código. El control de versiones también ayudará al profesor a llevar un control sobre el desarrollo de la práctica. Se espera que las diferentes registros dentro del control de versiones sean cambios graduales. En caso contrario, se procederá a realizar un escrutinio con el objetivo de evitar fraudes.