Computability

Computable functions

What does it mean for a function to be computable?

Informally:

> A function is computable if we can define a WHILE program that computes it.

$$f : \mathbb{N} \longrightarrow \mathbb{N}$$

f is a partial function from $\mathbb{N}$ to $\mathbb{N}$

↓ accommodates the spinning of WHILE programs

A WHILE program S computes $f : \mathbb{N} \to \mathbb{N}$ with respect to x if $f(m) = n$ exactly when

$$\langle S, [x \mapsto m] \rangle \xrightarrow{*} [x \mapsto n]$$

- Our WHILE program...
- ... with distinguished variable x initialised to value m, and all other values set equal to zero ...
- many step evaluates to ...
- ... a state where all variables are equal to zero, except x whose value is the desired result $n : \mathbb{N}$

$$[\![S]\!]_x : \mathbb{N} \longrightarrow \mathbb{N}$$

Examples

① The function

$$f : \mathbb{N} \to \mathbb{N}$$
$$f(x) = x + 1$$

is computable.

We demonstrate this with the following WHILE program that computes f with respect to x

↓ "wrt"

Alternative statement:

We demonstrate this with the following WHILE program, S, such that $f = [\![S]\!]_x$

$$S = x \leftarrow x + 1$$

② The factorial function $(!) : \mathbb{N} \to \mathbb{N}$ is computable

We demonstrate this with the following WHILE program that computes $(!)$ wrt x:

```
r ← 1;
while (x ≥ 1) {
    r ← r * x;
    x ← x - 1;
}
x ← r;
r ← 0;
```

Q Show that the function $fib : \mathbb{N} \to \mathbb{N}$ is computable where

$$fib(n) = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ (n-1) + (n-2) & \text{otherwise} \end{cases}$$

aka it calculates the n^{th} fib num.

Summary of discussion

- we want a WHILE program
  - we must zero out our vars at the end.
  - we must name a distinguished var.
- should it terminate

  maybe? cos we want to calc fib

  BUT

  $$fib : \mathbb{N} \longrightarrow \mathbb{N}$$

  ↑ our definition of computability allows for non-termination.

④ The following partial function is computable.

$$g : \mathbb{N} \to \mathbb{N}$$
$$g(x) = \begin{cases} x + 1 & x < 2 \\ \uparrow & \text{otherwise} \end{cases}$$

We demonstrate this with the following WHILE program that computes g wrt x

```
if (x ≤ 1) {x ← x + 1;}
else {while true {skip};}
```

Characteristic Functions

For a given predicate on $\mathbb{N}$, its characteristic function returns 1 if the predicate holds for the given n, and 0 otherwise.

Formally:

Given a predicate on $\mathbb{N}$

$$M \subseteq \mathbb{N}$$

Its characteristic function is the function:

$$\chi_M : \mathbb{N} \to \mathbb{N}$$

$$\chi_M(n) = \begin{cases} 1 & \text{if } n \in M \\ 0 & \text{if } n \notin M \end{cases}$$

NOTE: these are always total

Decidable Predicates

A predicate $M \subseteq \mathbb{N}$ is decidable just if its characteristic function $\chi_M : \mathbb{N} \to \mathbb{N}$ is computable.

We call the WHILE program that computes χ_M a decision procedure.

Any predicate for which there is no decision procedure is called undecidable

Example

(1) The predicate $E = \{n \in \mathbb{N} \mid n \text{ is even}\}$ is decidable.

We will demo this with this WHILE program that implements it decision procedure wrt x:

```
while (x≥2) {
    x ← x-2;
}
x ← x+1;
```

This should be:

$x \leftarrow 1 - x;$