# COMPUTABLE AND RECURSIVE FUNCTIONS, AND CHURCH'S THESIS

What does it mean for a function $f : \omega \to \omega$ to be computable? This is a hard question to answer. Intuitively, it should mean something like this: There is a fixed procedure, such that given a natural number $n$, one can carry out this procedure in a finite amount of time with input $n$, and determine $f(n)$.

For instance, most people feel that the function $f(x) = x^2 + x + 1$ is computable, because given $n$, we first compute $n^2$ by adding $n$ to itself $n$ times (which requires further subtasks, perhaps), and then we add $n$ to it, and then we add 1, and this is the answer, $f(n)$. Of course, one must investigate why it is we believe that adding $n$ to itself $n$ times is a computable task, etc., which leads one to go back to how we were taught to add numbers in grade 1. At any rate, everyone believes this function $f$ is computable.

Several attempts have been made to formalize the notion of a computable function. Two such are Church's *Lambda calculus*, and Turing's *Turing machines*. A third possibility is the notion of a recursive function, given below. It turns out that all these approaches define the same concept (though they look rather different), which is seen as evidence that they all correctly formalize the (same) intuitive concept of a computable function.

**Primitive recursive and recursive functions.** We start by describing two schemes for creating new functions from old. We make the convention that a function $f : \omega^0 \to \omega$ is identified with an element of $\omega$. Alternatively, it can be thought of as a constant function.

Composition scheme. Given $h : \omega^m \to \omega$ and $g_1, \ldots, g_m : \omega^k \to \omega$, we may define a new function $f : \omega^k \to \omega$ by

$$f(x_1, \ldots, x_k) = h(g_1(x_1, \ldots, x_k), g_2(x_1, \ldots, x_k), \ldots, g_m(x_1, \ldots, x_k)).$$

Primitive recursion scheme. Given $k \geq 1$, and functions $h : \omega^{k-1} \to \omega$, $g : \omega^{k+1} \to \omega$, we can define a new function $f : \omega^k \to \omega$ by letting

$$f(0, x_1, \ldots, x_k) = h(x_2, \ldots, x_k)$$

and

$$f(x_1 + 1, x_2, \ldots, x_k) = g(x_1, f(x_1, \ldots, x_k), x_2, \ldots, x_k)$$

**Definition 0.1.** (I) The class of *elementary* functions consists of:

(1) The successor function $S : \omega \to \omega$, defined by $S(x) = x + 1$.
(2) The projection functions, defined for each $n \geq 1$ and $1 \leq i \leq n$ by $I_i^n(x_1, \ldots, x_n) = x_i$.
(3) All constant functions, i.e. $c : \omega^n \to \omega$ where $c(x_1, \ldots, x_n) = k$ for some $k \in \omega$, independently of $x_1, \ldots, x_k$.

(II) The class of *primitive recursive functions* is the smallest class of functions which contains the elementary functions and which is closed under the composition scheme and the primitive recursion scheme.

A more descriptive way of defining the class of primitive recursive functions is the following: A function $f$ is primitive recursive if only if there is a finite sequence $f_1, \ldots, f_n$ of functions with $f_n = f$, and where each $f_i$ is either elementary, or obtained by applying one of the schemes to functions that comes *before* $f_i$ on the list. (Exercise: Check this!)

In order to define the *recursive* functions, we need one more scheme.

$\mu$-OPERATOR SCHEME. If $g : \omega^{k+1} \to \omega$ is a function for which it holds that

$$(\forall x_1, \ldots, x_n \in \omega)(\exists y \in \omega) g(y, x_1, \ldots, x_n) = 0$$

then we may form a new function $f : \omega^n \to \omega$ by

$$f(x_1 \ldots, x_n) = \mu y[g(y, x_1, \ldots, x_n) = 0]$$

where $\mu y[...]$ means "the least $y$ such that [...].

**Definition 0.2.** The class of recursive functions is the smallest class of functions which contains the elementary functions, and which is closed under the composition scheme, the primitive recursion scheme, and the $\mu$-operator scheme.

Note that all primitive recursive functions are recursive. An alternative description of the class of recursive functions is: A function $f$ is recursive iff there is a finite list $f_1, \ldots, f_n$, where $f = f_n$, and where each element on the list is either an elementary function, or obtained by applying one of the three schemes to functions appearing earlier on the list.

There are recursive functions that are not primitive recursive, but that is a story for another day.

**Definition 0.3.** (1) A relation $R \subseteq \omega^n$ is recursive iff its characteristic function

$$\mathbf{1}_R(x_1, \ldots, x_n) = \begin{cases} 1 & \text{if } (x_1, \ldots, x_n) \in \mathbb{R} \\ 0 & \text{otherwise} \end{cases}$$

is recursive.

(2) A set $A \subseteq \omega$ is *recursively enumerable* if there is a recursive function $f : \omega \to \omega$ with $\text{ran}(f) = A$.

**Computable vs. recursive functions.** All recursive functions are computable (in the intuitive sense). This can be seen by induction: All elementary functions are clearly computable; if a function is formed by one of the schemes from computable functions, then it is also computable (if you don't see it, go back and re-read the schemes and convince yourself.)

In 1936, American logician Alonzo Church (1903–1995) introduced a class of computable functions in a different way, using what he called $\lambda$-calculus. It turns out that the class of recursive functions and the class of functions defined using $\lambda$-calculus are the same. They are also the same as the class of Turing computable functions, introduced by Alan Turing (1912–1954) in 1936 using an idealized notion of a computer. Church hypothesized the following:

**Church's thesis.** *The class of intuitively computable function on the natural numbers corresponds exactly to the class of recursive functions.*

This claim cannot be proved, since the notion of intuitively computable function is not a mathematical notion, but rather an intuitive idea. Church's thesis could potentially be *disproved* by giving and example of an intuitively computable function which is not recursive, but no-one has ever succeeded in doing so. Rather, the vast majority of mathematicians familiar with mathematical logic believe Church's thesis to be true.

**Connections to the book's notion of recursive and computable function.** Our book declares that a relation is recursive iff it is representable in a consistent finitely axiomatizable theory in a language containing the symbols $\mathbf{0}$ and $\mathbf{S}$. It further declares that a function $f : \omega^n \to \omega$ is recursive iff its graph, considered as a subset of $\omega^{n+1}$, is recursive.

It turns out that this definition is equivalent to the above definition (we may in fact see this before the end of the course). But it seems like a very strange definition of a concept that is supposed to capture an intuitive notion, which is why I chose to give the usual[1] definition of recursive function in class and above.

[1]By saying it is the usual definition, I mean that is the definition given in the vast majority of textbooks.