Let's begin with a definition:

Definition 1. A function $f : \mathbb{N}_0 \to \mathbb{N}_0$ is said to be computable if there exists an algorithm M that computes f. The algorithm M:

(a) halts on the input n and outputs $f(n)$ when f is defined for an $n \in \mathbb{N}_0$.

(b) does not halt on the input n whenever f is undefined for $n \in \mathbb{N}_0$.

Proof. (Not Really)

Let $f : \mathbb{N}_0 \to \mathbb{N}_0$ be a partial computable function. Extend f to a total function

$$f(n) = \begin{cases} f(n) & \text{if } n \in \operatorname{Dom} f \\ 0 & \text{otherwise} \end{cases}$$

The algorithm M does not halt on $n \notin \operatorname{Dom} f$. However, M is the only things we have to decide if $f(n)$ is defined. How do we know when to halt M and output $g(n) = 0$? □

Decidable Sets

Definition 2. A set $X \subseteq \mathbb{N}_0$ is said to be decidable if there exists an algorithm M that determines if $n \in \mathbb{N}_0$ belongs to X

Q: What does this mean?

A: The characteristic function for X

$$\chi_X(n) = \begin{cases} 1 & \text{if } n \in X \\ 0 & \text{otherwise} \end{cases}$$

is computable.

Remark 1.

(a) Any finite set is decidable. We can make an algorithm that outputs 1 on a finite number of elements.

(b) The collection of decidable sets is closed under intersection, union, and set difference.

Enumerable Sets

Any set that satisfies any of the following equivalent conditions is said to be <u>enumerable</u>.

Proposition 1. *Let* $X \subseteq \mathbb{N}_0$. *TFAE:*

(a) There exists an algorithm M that outputs the elements of X and only those elements.

(b) X is the domain of a computable function.

(c) X is the range of a computable function.

(d) The semicharacteristic function for X

$$f_X(n) = \begin{cases} 0 & \textit{if } n \in X \\ \textit{undefined} & \textit{otherwise} \end{cases}$$

Proof. (a) $\Rightarrow$ (b,d) Suppose that X is enumerated by an algorithm M. Then we can compute the semicharacteristic function for X:

(i) Execute M until n is printed.

(ii) Output 0 and terminate.

(c) $\Rightarrow$ (b) Suppose X is in the domain of a computable function f and N is the algorithm that computes f. We'll run N in parallel on 0, 1, 2, ... gradually increasing the number of involved inputs:

(i) Run one step of N on 0 and 1.

(ii) Run two steps of N on 0, 1, and 2.

(iii) Run three steps of N on each input 0, 1, 2, and 3.

All arguments on which N terminates are printed out as soon as they are detected. We avoid having N go into an infinite loop on an undefined element $n \in \mathbb{N}_0$ by only performing a finite number of steps from the algorithm N each time. We can modify this algorithm to print the results returned rather than the arguments where it terminates to prove the converse (b) $\Rightarrow$ (c).

(b) $\Rightarrow$ (c) We've seen that X is the domain of a computable function. This function is computed by an algorithm N. Define

$$b(x) = \begin{cases} x & \text{if } M \text{ terminates on } X \\ \text{undefine} & \text{otherwise} \end{cases}$$

$\square$

Definition 3. A set $X \subseteq \mathbb{N}_0$ is enumerable if either:

(i) $X = \varnothing$

(ii) X is in the range of a total computable function.

This definition turns out to be equivalent to all of the others. Suppose X is enumerated by an algorithm M. Let $x_0 \in X$ be arbitrary. Define the total function

$$a(n) = \begin{cases} t & \text{if } M \text{ returns } t \text{ at the } n^{\text{th}} \text{ step} \\ x_0 & \text{otherwise} \end{cases}$$

The function $a(n)$ has X in its range and is computable by its definition. Hence, X is enumerable.

Theorem 1. *The intersection and union of enumerable sets is enumrable.*

Remark 2. The complement of an enumerable set may not be enumerable.