

A Computability Problem

universality and limits in programmability – Part I

Start with a fact familiar from your discrete mathematics course.

FACT 1. *The set of all functions $\mathbb{N} \rightarrow \mathbb{N}$ is uncountably infinite.*

PROOF.

Let \mathcal{F} be the set of all functions $\mathbb{N} \rightarrow \mathbb{N}$.

Suppose: \mathcal{F} is *countably* infinite.

{

Let (f_0, f_1, \dots) be a listing of \mathcal{F} .

Define

[i] $d(n) = f_n(n) + 1$

Since d is also a function $\mathbb{N} \rightarrow \mathbb{N}$, it must appear in the list (f_0, f_1, \dots) somewhere.

Say it occurs in position k : $d = f_k$. That means

[ii] $d(n) = f_k(n)$ for all n .

Now apply $d()$ to the input k :

$d(k) = f_k(k)$ because of **[ii]**

But also: $d(k) = f_k(k) + 1$ because of **[i]**

Contradiction.

}

So \mathcal{F} is *not* countably infinite.

\mathcal{F} is clearly not finite. So it is uncountably infinite.

QED.

A function $f: \mathbb{N} \rightarrow \mathbb{N}$ is *effectively computable* if there exists a program that computes it. We may consider a “program” to be a sequence of ASCII characters that is a legal C program with signature **int f(int)**, with the understanding that there is no size limit on the integers represented by **ints**¹. The set of programs is countably infinite (since there is no more than a countable infinity of ASCII strings), so one can conclude:

FACT 2. *The set of all effectively computable functions $\mathbb{N} \rightarrow \mathbb{N}$ is countably infinite.*

Now consider the following line of reasoning, stolen nearly verbatim from the proof of Fact 1:

Let (f_0, f_1, \dots) be a listing of the (countable!) set of all effectively computable functions².

Define

[i] $d(n) = f_n(n) + 1$

Since d is also an effectively computable function $\mathbb{N} \rightarrow \mathbb{N}$, it must appear in the list (f_0, f_1, \dots) somewhere.

Say it occurs in position k : $d = f_k$. That means

[ii] $d(n) = f_k(n)$ for all n .

Now apply $d()$ to the input k :

$d(k) = f_k(k)$ because of **[ii]**

But also: $d(k) = f_k(k) + 1$ because of **[i]**

Contradiction.

What?! Did we just prove that the set of effectively computable functions is *uncountable*? That’s false!
What’s causing the problem here?

¹ By standard results covered in Theory of Computation courses, for “C program” you can substitute your favorite programming language, your favorite flavor of Turing machines, formal grammars, mu-recursive functions, etc. , and there will be no difference.

² You might think of ways to produce this listing. Suppose someone gives you a number n ; you can get f_n this way: start generating all ASCII strings, in order of increasing length. When you generate each string, apply a C syntax verifier and see if it happens to be legal code for a function **int f(int)**. If so, increment a counter, and go on to the next string. When this counter hits n , you have f_n .