COMMON PROOFS

James Thomas Parker, MS

Post Graduate

NCSU Raleigh, N.C.

910-798-3052

September 17, 2017

KEYWORDS: Non-computability, Komolgorov String, Complexity, MAXCLOCK, Halt Function

Abstract:

Three common proofs that that are generally accepted as true are shown in this paper to have flaws in them. This paper does not attempt to resolve these flaws but it does

request a peer review of their conclusions. Other proofs will be needed to maintain their truth. Math and Logic do not lie but no one wants to take the risk of publishing these discussions. I invite you to read this paper and convince yourself of it's veracity

Introduction:

A leading professor at Stanford told me that the non-computability of the Halt function is as unassailable as logic itself. He may be correct but in light of my attempts to find a proof of its non-existence, no one has provided me with a single one. This is frustrating to me because I am only looking for the truth. In the days of Copernicus, people staunchly adhered to the belief that the sun went around the earth. They even persecuted people that claimed otherwise. I am not eagerly awaiting persecution but I do have the courage to learn what is actually true. If any reader can provide a solid reason for the conclusions of these proofs, I will welcome their input. I do not accept reasons like some authority said it is true. The reason must come from sound logic.

1. First Proof Concerning The Halt Function:

The first proof[4] that the Halt Function is non-computable is confusing. We have a Halt Function H such that if its argument halts, H runs forever and if its argument runs forever, H halts. The proof given assumes H2 calls H1; thus, we have the strange function H2(H1). Assume H1 goes in an infinite loop and H2 halts it. This seems like a paradox but it is actually not. If MAXCLOCK can be computed, all tests of computable functions have a MAXCLOCK, (MC) which if the number of clocks of the function exceeds it, it must run forever. We will show later that the existence of non-computable functions was proved by Turing so we cannot say MC is always computable. We will look at that proof later. Let H1 be in an infinite loop so its clocks exceed MC. H2 can let H1 exceed MC by 1 clock and then halt H1 since it takes at least 1 clock to test if MC was exceeded. Thus it can state H1 runs forever and halt H2. No contradiction in the functions of H.

2. Second Proof Concerning Komolgorov Complexity:

The Komolgorov complexity of a string is assumed to be non-computable. A proof of that is shown here to have a subtle flaw.[3][2]

We provide a simple proof by contradiction that the Komolgorov complexity of a string is non-computable. It was found on a standard web site:

function KolmogorovComplexity(string s)

takes as input a string s and returns K(s). Now, consider the program

```
function GenerateComplexString(int n)
   for i = 1 to infinity:
      for each string s of length exactly i
         if KolmogorovComplexity(s) >= n
            return s
```

Given any positive integer n, it produces a string with Kolmogorov complexity at least as great as n. The program itself has a fixed length U. The input to the program

GenerateComplexString is an integer n. Here, the size of n is measured by the number of bits required to represent n, which is log2(n). Now, consider the following program:

```
function GenerateParadoxicalString()
      return GenerateComplexString(n0);
```

This program calls GenerateComplexString as a subroutine, and also has a free parameter n0. The program outputs a string s whose complexity is at least n0. By some choice of the parameter n0, we will arrive at a contradiction. To choose this value, note that s is described by the program GenerateParadoxicalString whose length is at most

U+log2(n0)+C

where C is the "overhead" added by the program GenerateParadoxicalString. Since n grows faster than $\log_2(n)$, there must exist a value n0 such that

U+log2(n0)+C<n0

But this contradicts the definition of s as having a complexity at least n0. That is, by the definition of K(s), the string s returned by GenerateParadoxicalString is only supposed to be able to be generated by a program of length n0 or longer, but GenerateParadoxicalString is shorter than n0. Thus the program named "KolmogorovComplexity" cannot actually find the complexity of all strings.

Problems With This Proof

This proof is flawed because U is not a constant. For small n0 we can say that U+log2(n0)+C > n0 but we cannot say what happens to the size of U as n0 gets larger. U may grow in size as n0 increases. The contradiction described above has not been proven.

3. Non-computable Functions:

Turing's proof that some functions are non-computable [1] is correct using logic. But this does not prove MAXCLOCK is non-computable. He uses a function we will call "f" which is a function of all functions, including itself. In his proof, he arrives at a contradiction and concludes that some of the functions of f must be non-computable. I agree with his conclusion. But how can we prove in our contradiction which of the functions f calls are non-computable? The only function he proved was non-computable was f itself. Because it calls f and all other functions, none of the other functions need be non-computable. We do not have a solid proof that the Halting Problem is non-computable.

.4: References:

[1] Computability_and_Complexity_Theory,_2nd_Edition_-Springer(2012).pdf <http://carlossicoli.free.fr/H/Homer_S.,_Selman_A.L.-> Accessed 28, Oct 2013.

[2] "Kolmogorov Complexity." Research paper from Department of Computer Science, University of Chicago. <http://www.cs.uchicago.edu/~fortnow/papers/kaikoura.pdf>. Accessed 23 Oct 2013.

[3] "Komolgorov Complexity." ***Iprogrammer***. <http://www.i-programmer.info/babbages-bag/6319-kolmogorov-complexity.html> Accessed 23 Oct 2013.

[4] Lecture 10: Unsolvability and The Halting Problem, Brooklyn College

<http://corc1312.ezirim.com/lectures/lecture-10/>
Accessed 28 Oct 2013.

[5] Parker, James Thomas. ***Mathshowcase.com***. <http://mathshowcase.com>. Accessed 17 Sept. 2017.