On Deciding β' from an $\mathscr{H}$ Machine Through a Self Validating Process

Mark Inman, Ph.D.

February 27, 2015

Abstract

This paper contains a description for the configuration of a self-validating Turing Machine. This description allows such a machine to solve for β'.

1

1.1 Introduction

This paper contains a description for the configuration of a self-validating Turing Machine. Solving for β' is the equivalent to finding an exception to diagonalization.

1.2 Preliminaries

The terms *Circular Machine* and *Circle-free Machine* are no longer used by convention. However, it is convenient to note here that a *Circular Machine* is unsatisfactory due to it's inability to continue based on the rules of a Universal Turing Machine, $\mathscr{U}$, set forth in Turing's Paper on the Entscheidungsproblem. In addition, a *Circle-free Machine* is satisfactory based on those same rules and an ability to continue enumerating a calculation set. [1]

A *Standard Description* or S.D. is the rule set for any given Turing Machine $\mathscr{M}$ in a standard form. By creating a standard, the rule sets themselves can be used to create a *Description Number* or D.N. which itself is readable by a Universal Turing Machine, $\mathscr{U}$, as an instruction set. [1]

From Turing's paper: "Let $\mathscr{D}$ be the Turing Machine which when supplied with the Standard Description (S.D.) of any computing machine $\mathscr{M}$ will test this S.D. and if $\mathscr{M}$ is circular will mark the S.D. with the symbol "u" and if it is circle free, will mark it with 's' for 'unsatisfactory' and 'satisfactory' respectively. By combining machines $\mathscr{D}$ and $\mathscr{U}$, we could construct a machine $\mathscr{H}$ to compute the sequence of β'" [1] $\mathscr{H}$ is circle free by construction since each section after the Turing machine moves to the next section, is itself finite and the calculations should take a finite number of steps. [1]

It is not in the scope of this paper to provide an in depth description of computable numbers and their processing. The only additional preliminary is that one accepts the Church-Turing thesis that says every function which is calculable, is a computable function. For an in depth understanding of Turing Machines, the Universal Turing Machine and the history behind it's development, please read *The Annotated Turing* by Charles Petzold. [1]

1.3 Turing's Claim

In the eighth section of Turing's paper on the Entscheidungsproblem, Turing claims that $\beta^{'}$ can not be determined vis a vis the following reason:

"The instructions for calculating the R(K)-th [figure] would amount to 'calculate the first R(K)-th figures computed by $\mathscr{H}$ and write down the R(K)-th'. This R(K)-th would never be found. I.e. $\mathscr{H}$ is circular..." [1]

This is because, since $\mathscr{H}$ relies on subroutines, when it reaches and tries to evaluate K, it must call itself, which provides instructions on reading inputs from 1 to K-1 in order to call the R(K)-th figure, but it can never get there, because it keeps repeating it's own instruction loop. [1] The question is, however, is there a Turing Machine which can recognize itself arbitrarily[1] when it reaches it's own Description Number (D.N.), such that $\mathscr{H}$ prints $\beta^{'}$?

2

2.1 Supermachine

Now, let us determine that $\mathscr{H}^{'}$ is a controller machine with a D.N. of $K^{'}$. It controls two different $\mathscr{H}$ machines: $\mathscr{H}_0$ and $\mathscr{H}_1$. $\mathscr{H}_0$ and $\mathscr{H}_1$ have the same ability to determine "u" or "s", except $\mathscr{H}_0$ tests as Turing describes, from 1 counting upwards and $\mathscr{H}_1$ tests tests from the twos complement of whatever number is being tested by $\mathscr{H}_0$ and subtracts one instead of adding one to find the next number. They take each input simultaneously until the controller machine finds a redundancy. They each have a unique D.N. of K_0 or K_1 through an identifier string which differentiates the two $\mathscr{H}$ machines from each other. The key requirements for $\mathscr{H}^{'}$ include storing output value pairs from the two connected $\mathscr{H}$ machines, recognizing redundancies and recognizing when a redundancy is met with an existing nil key value on the input.

Let $\mathscr{H}_s$ be the supermachine that is the combination of all three $\mathscr{H}$ Machines and K_s is the D.S. for the supermachine.

Initialize the identifier strings such that $K_1 < K_0$.

Let the number of bits in the twos complement be determined by n number of bits in K_0.

Let the twos complement be the number $c = 2^n - 1$, If $c - K_0 > K_1$, then re-initialize the identifier string in either K_0 or K_1 such that $c - K_0 < K_1$. This guarantees that $\mathscr{H}_0$ will read K_1 before $\mathscr{H}_1$ reads K_1 and also guarantees $\mathscr{H}_1$ will read K_0 before $\mathscr{H}_0$ reads K_0. When the machines have determined the field, let the controller $\mathscr{H}^{`}$ contain a controller function which stores the value pairs of $\mathscr{H}_0$ and $\mathscr{H}_1$, checks for a redundancy on the input and proceeds to utilize machine $\mathscr{H}_0$ to procede upwards from $c + 1$.

[1]by recognizing itself arbitrarily, we mean that it can recognize itself even if K is dynamic and changes as the program continues to run, or there is no initializer that feeds a static K to be recognized by a sngle read instruction that skips K and just rubber stamps apporval.

At this point it should be noted that the $\mathscr{H}^{'}$ controller takes it's instructions from $\mathscr{H}_0$ and $\mathscr{H}_1$.

2.2 $\beta^{'}$ is Decidable

Proof. At the point $K^{'}$ is received as an input, it is determined satisfactory by either $\mathscr{H}_0$or $\mathscr{H}_1$ without calling either machine.

Finally, K_s, which describes the super machine $\mathscr{H}$ is read by $\mathscr{H}^{'}$ and like all other strings, is stored with a value of nil until $\mathscr{H}_0$ or $\mathscr{H}_1$ returns a value for $\beta^{'}$ at that location. K_s is sent to be verified by $\mathscr{H}_0$, which when $\mathscr{H}^{'}$ calls K_s under the iteration, is recognized as redundant by $\mathscr{H}^{'}$ in the data store, but because th value is already written as null, the controller stores the fact this process occurred, sends the same K_s to $\mathscr{H}_1$, which verifies the redundancy and nil value. $\mathscr{H}^{'}$ now self-verifies the input K_s as it's current operating D.N., provides a value of "s" and moves to evaluate section $K_s + 1$.

Therefore, $\beta^{'}$ is decidable for any Description Number given some $\mathscr{H}$ Machine. ∎

References

[1] Charles Petzold *The Annotated Turing* 2008: Wiley Publishing, Indianapolis, IN.