

Re: What is the Result from Invoking this Halt Function?

Source: <http://sci.tech-archive.net/Archive/sci.logic/2004-08/2465.html>

From: Tim Peters (*tim.one_at_comcast.net*)

Date: 08/19/04

Date: Wed, 18 Aug 2004 23:00:17 -0400

[>parr(*>)]

> ...

> *The really bad news is that the part of the paper which most closely*

> *corresponds to the halting problem is not considering circular nor*

> *circle-free.*

I don't think so. The unsolvability of what's now called "the halting problem" for the machines in the paper is a very small step if you take as given the earlier proof that no machine can determine whether an arbitrary machine ever prints a 0. That latter result is also the key to the later proof of the unsolvability of the decision problem, but can be applied directly to the halting problem.

For example, assume a halt-detector H exists. Take the standard description of any machine M. Derive a machine M' by replacing the transition on each (if any) "print 0" step in M with a transition to the halting state. Run H on the description of M'. If H says M' doesn't halt, conclude that M never prints a 0. If H says M' does halt, run M' to completion and report on whether it ever printed a 0. Now you've got an effective process for determining whether M ever prints a 0, contrary to that no such process exists, so H cannot exist either.

Given the huge number of details Turing left out in the paper, and how straightforward that demonstration is, I doubt Turing would have bothered writing it down, even if he had thought it an interesting result.

Gloss: Showing that M' prints a 0 iff M prints a 0 is one of the kinds of details Turing routinely left out, presumably because they were too obvious to him to bother noting. It depends in part on that the flavor of machine defined in the paper cannot (intro)inspect its own m-configuration (if it could, then M could have steps of the form "if my state #78643 has a transition to the halting state, transition to state #78644 else to #42", and deriving M' from M while preserving relevant behaviors could be a tedious undertaking). The numbing febleness of the Turing machine's repertoire makes some kinds of proof techniques easier than in seemingly

sci.logic: Re: What is the Result from Invoking this Halt Function?

richer models of effective computation. For example, the only way for M to print a zero is to have a "print 0" step — there are no variable bindings, no indexed lookups, no clever recursions to worry about. M has steps to print 0 or it doesn't, they're trivial to find, and trivial to transform without changing the behavior of the machine on any paths from the start state to their states.