

Re: why is halting problem profound?

Re: why is halting problem profound?

Source: <http://sci.tech-archive.net/Archive/sci.logic/2008-03/msg02045.html>

- *From:* Mitch <maharri@xxxxxxxx>
 - *Date:* Thu, 27 Mar 2008 19:58:25 -0700 (PDT)
-

On Mar 27, 9:29 pm, shimp <exam...@xxxxxxxx> wrote:

Hi. Recently learned of the 'halting problem'. Im no philosopher (but I know computers). I'm wondering why this theorem is considered to be so profound and celebrated?? No offense to Turing, a great mind.

The theorem 'the halting problem is unsolvable' is important because it says that there are some things you just –cannot– do. It's easy enough to think up things one might like to do (find the greatest common divisor of 2 numbers, color a map with 4 colors, determine if white can guarantee a win in chess, etc, etc). And some of these have answers (that is humans have come up with a solution (an algorithm that terminates correctly on all inputs), or have not yet figured a solution). But the thm gives an example of some problem (the halting problem in particular) that does not have a solution. Before the thm, people thought it was plausible that there might be an algorithm to solve any problem posed to it (but humans just hadn't thought of one so clever yet). (notice the mix of ideas here that is not really a mix: an algorithm that takes an algorithm as input)

My end analysis is that he is basically saying, you cannot determine if ANY and ALL possible loops of code will ever end, unless you execute the loop. Whether you actually execute the code, or you unroll it using some interpreter.. either way you are executing it. It may never end, so you may never have an answer.

Is this non-obvious to some people? The wording of the theorem is so abstract and it says nothing about the kind of code or instructions that the machine is allowed to execute. It could do, or be, anything. No way you can know until you run it.

Did I miss something? Maybe I interpreted it wrong?

lots of points here:

– "is this non-obvious to some people?" (this being "unless you

Re: why is halting problem profound?

Re: why is halting problem profound?

execute the code, you can't determine if all loops will end"

No it's not obvious to some. (consider a Java like formalism rather than TMs for simplicity). Most loops in programs you see – obviously– terminate: they have simple bounds. But some loops might have strange bounds on them that are variables that change both up and down, and it is not obvious how they change (consider Euclid's GCD, yes the guard variable always decreases, but it's not obvious by looking at the guard, you have to take the (additional very short obvious) step of noticing that $a-b < a$ (see any implementation of GCD). Some programs have much more bizarre guards on their loops, which are even less obvious.

– "The wording of the theorem is so abstract and it says nothing about the kind of code or instructions that the machine is allowed to execute."

That depends under what circumstances you see the theorem presented. The presentation should probably define some formalism of process (some variant of a Turing Machine). The TM operations are exactly the very concrete instructions (code if you like) that are executed. It may sound abstract because there are no 'real' machines that act that way (actually I bet all sorts of TM 'compilers' have been made), but then any machine language for a given chip, or any programming language is then "so abstract".

So, executive summary: 'The halting problem is unsolvable' is profound and uncelebrated because it is definite knowledge that there are limits to...well...knowledge. And that (technical) fact is not obvious either. (one could use it the idea in the humanities as a metaphor, just like the relativity theory, but then that would ignore the technicalities of how the original thm is proved/used.

Mitch

.