

Re: why is halting problem profound?

## Re: why is halting problem profound?

---

*Source:* <http://sci.tech-archive.net/Archive/sci.logic/2008-03/msg02049.html>

---

- *From:* Marshall <marshall.spight@xxxxxxxxxx>
  - *Date:* Thu, 27 Mar 2008 22:21:39 -0700 (PDT)
- 

On Mar 27, 5:29 pm, shimp <exam...@xxxxxxxxxx> 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.

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.

Right: unless you execute the code, AND it happens to halt. If it doesn't halt, you don't know anything.

Whether you actually execute the code, or you unroll it using some interpreter.. either way you are executing it.

Even if you're not executing it. If you have some fancy abstract interpretation technique, or some kind of source code analysis, some type theory, whatever. The general case of the problem is not decidable.

It may never end, so you  
may never have an answer.

Right.

Is this non-obvious to some people?

It is non-obvious to everyone.

Re: why is halting problem profound?

Re: why is halting problem profound?

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.

Yes! That's part of what's so profound about it. It doesn't depend on the nature of the computational model!

It could do, or be, anything. No way you can know until you run it.

Did I miss something? Maybe I interpreted it wrong?

Another thing about the halting problem is that it sits "right next to" a whole bunch of other interesting results. For example the busy beaver problem.

One way we could try to solve the halting problem would be with an interpreter. Plug the TM into the interpreter, and if it runs for long enough without halting, then conclude it's in an infinite loop. (And of course, some TMs do enter an infinite loop.) So why doesn't that work? Because you can't put a bound on how much time to allow.

One might respond that one simply needs to come up with a calculation, parameterized on the number of states the TM has, that is greater than the largest number of steps the machine can execute before it's definitely in a loop. It doesn't even have to be any kind of optimum; the least upper bound is not necessary. ANY upper bound will do. But it turns out this is impossible \*because that function grows too fast.\* In other words, you have a function that is uncomputable because it just gets too big to quickly. That, to me, is a pretty amazing fact.

Marshall

.

Re: why is halting problem profound?