

Re: Disproof of the Halting Problem's Conclusion

Source: <http://sci.tech-archive.net/Archive/sci.logic/2004-07/2627.html>

From: Chris Menzel (cmenzel_at_remove-this.tamu.edu)

Date: 07/25/04

Date: 25 Jul 2004 05:27:49 GMT

On Sun, 25 Jul 2004 04:21:34 GMT, Peter Olcott <olcott@worldnet.att.net> said:

>
> *"Chris Menzel" <cmenzel@remove-this.tamu.edu> wrote in message*
> *news:slrncg68gf.5rf.cmenzel@philebus.tamu.edu...*
>> *On Sat, 24 Jul 2004 13:54:56 GMT, Peter Olcott <olcott@worldnet.att.net> said:*
>> >
>>> *"Chris Menzel" <cmenzel@remove-this.tamu.edu> wrote in message*
>>> *news:slrncg3s7d.5rf.cmenzel@philebus.tamu.edu...*
>>>> *This is ill-defined rubbish. Turing's proof has nothing to do with*
>>>> *results being returned to "the program being analyzed". If you*
>>>> *understood the proof and the mathematics behind it, you'd see that this*
>>>> *is so.*
>>> >
>>> *Introduction to Formal Languages and Automata by Peter Lintz page 319.*
>>>
>> *Ah, this would be the same text in which Lintz proves that the Halting*
>> *Problem is unsolvable? Why is it you don't take *that* as*
>> *authoritative?*
> //
> //DO NOT CHOP THIS OUT
> //
> +-----+
> ||
> v |
> q(0)----->q(y)----->q(a)----->q(b)
> |
> |
> +----->(q(n))
>
>>> *The state transition from q(y) to q(a) has the same meaning as*
>>> *returning the result of the halt analysis to its caller, since q(y) is*
>>> *the result of the halting analysis, and q(a) is (equivalent to) the*
>>> *caller.*
>>>
>> *You've given no indication you have the slightest idea what a state*
>> *transition is. And this passage is so full of undefined terms (q(y)?*
>> *q(a)?) that it is meaningless.*
>

sci.logic: Re: Disproof of the Halting Problem's Conclusion

- > *It sure is meaningless when you chop out the definition of my terms.*
- > *I had to go back a search through all the messages to restore what*
- > *you chopped out.*

You are confused. I quoted everything you had written in the post to which I was responding. There were no definitions of $q(y)$ etc. (And there still aren't in the skeleton you provide above -- not that it really matters vis-a-vis the issue at hand.)

- >> *That said, I won't go so far as to say that there is no programming*
- >> *language version of the Halting Problem on which it makes sense to talk*
- >> *about calling and callers. Maybe there is. The point, however, is*
- >> *that, for some reason, you've confused yourself into believing the*
- >> *following piece of utter nonsense:*
- >
- > *That last statement was rhetoric rather than refutation, can't you stick*
- > *with more exacting terms?*

I have, and will continue to do so. The terms in this proof of the unsolvability of the Halting Problem are completely precise:

<http://tinyurl.com/6ow6c>

- >> > *Like I have said many times, Turing did not prove that solving*
- >> > *the Halting Problem is impossible. All that he really showed*
- >> > *was one way that can not possibly work in every case.*
- >>
- >> *But you are completely, pathetically, provably wrong. Here is what*
- >
- > *You are much better with rhetoric than you are with reasoning,*

I provided you with a very precise piece of reasoning, which, everyone has noticed, you studiously avoid.

- >> *Turing proved (have a look at that Lintz text): There is no Turing*
- >> *machine (alternatively, no program in a Turing complete programming*
- >> *language) that computes the halting function. Period. Note that says*
- >> *NOTHING about some particular *way* of solving it. You see that, don't*
- >> *you? Here's the proof: <http://tinyurl.com/6ow6c> . So you are OBVIOUSLY*
- >> *wrong. You have OBVIOUSLY not shown what you think you've shown. It*
- >> *is, alas, obvious to everyone but you.*
- >
- > *Try to show your refutation in terms of this example*
- > <http://www.netaxs.com/people/nerp/automata/halting2.html>
- > *If this example (used for a college course on the subject) is*
- > *woefully inaccurate, try to show where and how and why it errs.*

Ok. Unfortunately, it is indeed rather inaccurate and unclear.

* Most problematically, it does not define what a Turing machine is. It does not define its inputs and outputs. It doesn't describe its actions. The version of the proof I provided gives a precise

definition.

* The version you provide uses the variable "M" initially to refer to a string, but then talks about "Turing machine M". I *guess* what this is meant to suggest is that Turing machine M is the TM defined by the string M, when the latter turns out to be a program rather than just a random string. But what is the alphabet from which these strings are given? What is the language in question? The version I provided defined quite precisely the little language used to define TMs.

* Finally, the critical function "Impossible" in the proof you prefer appears to involve a use/mention problem. The function LoopIfHaltsOnItself appears to take that function itself as input. But that can't be right, as the relevant inputs here are supposed to be strings. So perhaps the idea is that LoopIfHaltsOnItself takes the string "LoopIfHaltsOnItself" as its argument. But that can't be right, since that string isn't a program. Presumably, the argument we need is the program itself; so what should have been written is something like

function Impossible:

```
return LoopIfHaltsOnItself ("function LoopIfHaltsOnItself (M):  
    return LoopIfHalts (M, M);");
```

So "LoopIfHaltsOnItself" in your version is being used both as a name for the function defined by the program in question and (illegitimately) as a name for the program itself qua string. This is a potential source of confusion, unclarity, and unfocused debate.

Granted, we could clean all of this up. But since the framework of your version is in fact much more complex than the framework of the version I posted, it would be a lot more difficult to do. That's why I posted a very basic, easy to understand version in which: Turing Machines are simple and well defined; "programs" for TMs are well-defined; names for TMs (viz., the number of each machine in the list of all machines) are well-defined; the inputs to TMs are well-defined, etc. There are also no use/mention problems.

So the version you want to use is sloppy and unclear, and therefore leaves room for error and misunderstanding. The simple, precise version I provide enables clear and focused discussion. It will therefore help you to make your points with that much greater force and clarity. So if you're interested in actually doing some mathematics, the version I provided, based on a widely-used and highly regarded text, is the one we need to use.

> *Since the state transition diagram version of the Halting Problem*
> *is analogous to this version,*

Analogy is for prose. We're doing mathematics here.

sci.logic: Re: Disproof of the Halting Problem's Conclusion

- > *I am estimating that both of these two versions are accurate*
- > *representations of the Halting Problem. If not, then please enlighten*
- > *me.*

See above. Now once again -- identify the flaw: <http://tinyurl.com/6ow6c>

Chris Menzel