

Re: The proof that I was referring to is on the website

Source: <http://sci.tech-archive.net/Archive/sci.logic/2004-08/0835.html>

From: Matt Chaplain (*nospam_at_kastagaar.com*)

Date: 08/06/04

Date: 6 Aug 2004 02:38:51 -0700

"Peter Olcott" <olcott@worldnet.att.net> wrote in message
news:<RspQc.390722\$Gx4.382955@bgtnc04-news.ops.worldnet.att.net>...

> "Matt Chaplain" <nospam@kastagaar.com> wrote in message
news:5e256172.0408050257.4cf19489@posting.google.com...

>> I think what the lads are trying to say is that a function, when run,
>> may do one of four things:

>>

>> 1) Return true

>> 2) Return false

>> 3) Return something else

>> 4) Not return.

>>

>> A Halt Analyser, by the very nature of the problem, is defined as
>> correctly returning true, should an input program halt, or correctly
>> returning false, should the input program not halt. These are its
>> only valid responses.

>

> and it can do this for every possible input

No, it can't. That's the point. The proof states that, given a hypothetical working Halt Analyser, we can contrive a scenario where the *input to the Halt Analyser* (to use your terminology) causes the Halt Analyser to be unable to produce correct results, ergo the hypothetical Halt Analyser wasn't a Halt Analyser.

As I stated in my last mail, if the "Conditions" under which the test is running causes the Halt Analyser to fail, then simply imagine a new Halt Analyser which works under these "Conditions", and reapply the proof.

For example, assume that your Halt Analyser works when invoked upon an algorithm containing any Halt Analyser. You can *still* apply the proof and find out that it doesn't.