

Re: Godel proved maths inconsistent not incompleteness theorem

Source: <http://sci.tech-archive.net/Archive/sci.logic/2008-05/msg00320.html>

- *From:* David C. Ullrich <dullrich@xxxxxxxxxxxx>
 - *Date:* Tue, 06 May 2008 04:59:28 -0500
-

On Mon, 5 May 2008 21:25:23 +0000 (UTC), Chris Menzel <cmenzel@xxxxxxxxxxxxxxxxxxxxxxxx> wrote:

On Mon, 5 May 2008 12:25:46 -0700 (PDT), Charlie-Boo <shymathguy@xxxxxxxx> said:

On Apr 30, 12:59 pm, Chris Menzel <cmen...@xxxxxxxxxxxxxxxxxxxxxxxx> wrote:

On Wed, 30 Apr 2008 01:54:34 -0700 (PDT), Charlie-Boo

Can you code from specs that contain "e.g."
and "etc."?

I'm not giving you a spec.

That's the problem. You don't even have a spec and you say it's trivial.

That's because a decent programmer can often tell when a task is trivial. It is no surprise, of course, that you are unable to see the triviality in question, given that you are still terribly confused about the distinction between theorem proving and proof checking.

That of
course
depends on
how wffs
are
represented.

Re: Godel proved maths inconsistent not incompleteness theorem

No kidding. Presumably, they are "represented" they way WFFs are typically "represented" in first-order languages, as strings in a certain recursively defined set. Standard stuff.

Then why do the various expositions use different syntaxes?

Uh, what? Different syntaxes still use strings of symbols. (Also, but for minor details (e.g., the presence of "0" as a primitive symbol), the different expositions use exactly the same syntax. It's the *axioms* they use that can differ.)

Zohar Manna and Richard Waldinger gave up trying to implement Inductive proofs years ago.

Inductive proofs? You think that is even *relevant* in a discussion of *deductive* proof checkers? Just how confused *are* you about the issues here?

ZF without Induction??

I understood you to mean "induction" in the sense of "inductive logic", which of course is extremely difficult to implement. Not that this matters in the least to the point. Even if we are talking about induction over sets and numbers in the deductive sense, you are still confusing proof checkers with theorem proving. The difficulties of implementing proof mechanisms for induction in either sense is simply irrelevant to the issue of proof checking.

And just FYI: ZF doesn't "have" induction, at least, not as an axiom. It is derivable.

As mentioned already, I gave you a link to a fully implemented proof checker for a complete system of natural deduction, one that,

Re: Godel proved maths inconsistent not incompleteness theorem

in particular, implements 20 of those rules of inference that you call "the hard part". All one would have to do to convert it into a dedicated proof checker for ZF, PA or what have you would be to implement some elementary pattern matching routines for the axioms/schemas of your chosen theory.

Oh, I see. It's trivial if you already have a program that does 90% of it.

Yes indeed, all of logical infrastructure including the rules of inference are coded. That you are trying to make an issue of the fact that the simple pattern matching routines for axiom checking have not actually been coded shows that you are either blatantly dishonest or a completely incompetent programmer.

1. For each line in the purported proof
 - a. Run one step of the theorem prover
2. For each theorem generated, compare it to the theorem at that line.
3. If equal, then go to the next line (1)
 - a. If no more lines, the proof is valid.
4. If no more theorems to generate, the proof is not valid.

So you think hard things are easy and easy things are hard.

There is a pretty serious problem with your algorithm, specifically the "If no more theorems to generate" condition in line 4 -- you *do* know there are infinitely many theorems of first-order logic, right?

There are a finite number of theorems at that single step.

Er, you really think that helps? Seriously? You do?

Re: Godel proved maths inconsistent not incompleteness theorem

Whether it helps or not, it's simply not true. At least not necessarily – there are plenty of formal systems for which it's false.

There are a number of popular axiomatizations of ZFC.

I thought you said that the syntax is standard?

I don't know what you are referring to, but "syntax" and "axiomatization" are not synonyms. Just FYI.

Anyone who has developed code of any complexity knows you can never tell how big a task is until you spec it out.

Sure, to some extent, but if you've got a firm grasp of your subject matter, some programming talent and some experience behind you can often tell that a given task is going to be routine. Writing a proof checker for example.

Like I said, this sort of thing comes from a blend of book learning, experience, and natural ability.

I thought it was trivial – why does it now take all that? Trivial if Andrew Wiles did it?

Re: Godel proved maths inconsistent not incompleteness theorem

More silly rhetorical tricks to obscure the fact that you have nothing to say. What was said to be trivial was the task of extending a proof checker with routines to determine whether a given sentence is an axiom of ZFC. What I am talking about above is the capacity to see *whether* a given programming task is trivial without actually programming it. I am not claiming that that capacity is trivial.

I have a hard time deciding whether you are simply monumentally muddle-headed or both muddle-headed (but not quite monumentally so) and dishonest.

David C. Ullrich