

Re: Aspiring highest-order programmer

Source: <http://coding.derkeiler.com/Archive/General/comp.programming/2004-06/0708.html>

From: Ian Woods (newspub2_at_wuggyNOCAPS.org)

Date: 06/12/04

Date: Sat, 12 Jun 2004 02:56:08 +0000 (UTC)

spinoza1111@yahoo.com (Edward G. Nilges) wrote in
news:f5dda427.0406111715.40a601fa@posting.google.com:

> Christopher Barber <cbarber@curl.com> wrote in message
> news:<psozn7arxca.fsf@unicron.curl.com>...
>> spinoza1111@yahoo.com (Edward G. Nilges) writes:
>>
>>> I have replied separately to your request that I describe my
>>> understanding of the applied theory of NP-completeness. It was
>>> probably a mistake to do so, since anything other than what on
>>> usenet corresponds to moronized rote memorization (cutting and
>>> pasting a Wikipedia) will be "deconstructed" in a Sophistical, but
>>> moronic, fashion, to "prove" that "he doesn't know what he's
>>> talking about", when I do.
>>
>> You have so far not yet proved that you understand what NP-complete
>> means. That is nothing to be ashamed of, it is a somewhat esoteric
>> concept for most programmers, but from what you have posted so far,
>> you don't seem to truly understand what it is about.
>
> You are playing a game that derives from the fact that during my long
> career, "intellectual" "property" considerations have transformed
> active knowledge into dead secrets in such a way that one can never,
> in an exchange like this, "prove" that he or she has human knowledge.

The concepts of NP, NP-complete and NP-hard can be found in a vast amount of freely available information, either through the power of that Internet thingummy, or by referring to the wealth of papers published on them over the last 40-ish years. They're so basic they're even taught to students studying software engineering and computer science. They're not exactly a trade secret.

> The reason is alienation. IP concerns are that knowledge becomes part
> of inventory and in this context, an unalienated individual making
> knowledge claims independent of the organization can always be
> renarrated as at worst not having "knowledge" and at worst as a
> charlatan.
>

comp.programming: Re: Aspiring highest-order programmer

- > *The interesting result is that mentoring disappears and is replaced by*
- > *ideological training and indoctrination. The interesting result is*
- > *that today's programmers repeat the mistakes of the past because their*
- > *elders aren't man enough any more to make knowledge claims lest some*
- > *suit or some fat woman from human resources "expose" the "hollowness"*
- > *of their claims.*
- >
- > *The corporate game is that you are "human resources" and you don't*
- > *understand the theory, therefore anything I say can be and will be*
- > *misread.*

You don't exactly help the situation by going that extra mile to make everything you say more difficult for your audience to understand.

<snip>

- >> *Of course, that would be expected if they are writing code that*
- >> *solves an NP-complete problem. If you mean that inexperienced*
- >> *programmers may sometimes use bad algorithms, I am sure I would*
- >> *agree, but you should not use the terminology "NP complete fashion"*
- >> *to describe this. NP completeness refers to a property of the*
- >> *problem domain, not the algorithm chosen to solve it.*

<snip>

- > *The academic meaning "this problem is solvable only by an algorithm*
- > *whose efficiency formula is nonpolynomial" has been transformed by*
- > *recent usage (and only usage) to mean that the PROBLEM is "np*
- > *complete" or "np" but the definition implies the existence of the*
- > *overcomplex algorithm.*

For NP, it means that there is no known polynomial time algorithm to solve the problem. For NP-complete (where all the NP problems in that set are 'equivalent'), the moment someone comes up with a polynomial time algorithm to solve any NP-complete problem, all NP-complete problems can be solved in polynomial time. (It would mean that someone had effectively proved that the set of NP complete problems is in the set of problems which can be solved in polynomial time... which would be one hell of an accomplishment, and worth at least a million dollars last time I checked.)

In other words, and back to your point, it's the /lack/ of existence of a polynomial time algorithm to solve the problem which makes the problem believed to be part of set of NP problems.

- > *In place of understanding you are instead demanding adherence to a*
- > *pragmatic terminology which is merely praxis.*

It's a formal definition which has been around since the 1960's and is well understood by well read (and, like me, not so well read) computer scientists and related people.

In place of incorrect usage of specific terminology, I for one would prefer correct usage of that terminology.

comp.programming: Re: Aspiring highest-order programmer

Ian Woods

--
There is no sig.