P=NP, relativisation, and multiple choice exams

1 August, 2009 in expository, math.IT | Tags: P=NP, randomness, relativisation

The most fundamental unsolved problem in complexity theory is undoubtedly the P=NP problem, which asks (roughly speaking) whether a problem which can be solved by a non-deterministic polynomial-time (NP) algorithm, can also be solved by a deterministic polynomial-time (P) algorithm. The general belief is that $P \neq NP$, i.e. there exist problems which can be solved by non-deterministic polynomial-time algorithms but not by deterministic polynomial-time algorithms.

One reason why the $P \neq NP$ question is so difficult to resolve is that a certain generalisation of this question has an affirmative answer in some cases, and a negative answer in other cases. More precisely, if we give all the algorithms access to an oracle, then for one choice A of this oracle, all the problems that are solvable by non-deterministic polynomial-time algorithms that calls A (NP^A), can also be solved by a deterministic polynomial-time algorithm algorithm that calls A (P^A), thus $P^A = NP^A$; but for another choice B of this oracle, there exist problems solvable by non-deterministic polynomial-time algorithms that call B, which *cannot* be solved by a deterministic polynomial-time algorithm that calls B, thus $P^B \neq NP^B$. One particular consequence of this result (which is due to Baker, Gill, and Solovay) is that there cannot be any *relativisable* proof of either $P = NP$ or $P \neq NP$, where "relativisable" means that the proof would also work without any changes in the presence of an oracle.

The Baker-Gill-Solovay result was quite surprising, but the idea of the proof turns out to be rather simple. To get an oracle A such that $P^A = NP^A$, one basically sets A to be a powerful simulator that can simulate non-deterministic machines (and, furthermore, can also simulate *itself*); it turns out that any PSPACE-complete oracle would suffice for this task. To get an oracle B for which $P^B \neq NP^B$, one has to be a bit sneakier, setting B to be a query device for a sparse set of random (or high-complexity) strings, which are too complex to be guessed at by any deterministic polynomial-time algorithm.

Unfortunately, the simple idea of the proof can be obscured by various technical details (e.g. using Turing machines to define P and NP precisely), which require a certain amount of time to properly absorb. To help myself try to understand this result better, I have decided to give a sort of "allegory" of the proof, based around a (rather contrived) story about various students trying to pass a multiple choice test, which avoids all the technical details but still conveys the basic ideas of the argument. This allegory was primarily for my own benefit, but I thought it might also be of interest to some readers here (and also has some tangential relation to the proto-polymath project of determinstically finding primes), so I reproduce it below.

— 1. P and NP students —

In this story, two students, named P and NP (and which for sake of grammar, I will arbitrarily assume to be male), are preparing for their final exam in a maths course, which will consist of a long, tedious sequence of multiple-choice questions, or more precisely true-false questions. The exam has a reasonable but fixed time limit (e.g. three hours), and unlimited scratch paper is available during the exam. Students are allowed to bring one small index card into the exam. Other than scratch paper, an index card, and a pencil, no other materials are allowed. Students cannot leave questions blank; they must answer each question true or false. The professor for this course is dull and predictable; everyone knows in advance the type of questions that will be on the final, the only issue being the precise numerical values that will be used in the actual questions.

For each student response to a question, there are three possible outcomes:

Correct answer. The student answers the question correctly.

False negative. The student answers "false", but the actual answer is "true".

False positive. The student answers "true", but the actual answer is "false".

We will assume a certain asymmetry in the grading: a few points are deducted for false negatives, but a large number of points are deducted for false positives. (There are many real-life situations in which one type of error is considered less desirable than another; for instance, when deciding on guilt in a capital crime, a false positive is generally considered a much worse mistake than a false negative.) So, while students would naturally like to ace the exam by answering all questions correctly, they would tend to err on the side of caution and put down "false" when in doubt.

Student P is hard working and careful, but unimaginative and with a poor memory. His exam strategy is to put all the techniques needed to solve the exam problems on the index card, so that they can be applied by rote during the exam. If the nature of the exam is such that P can be guaranteed to ace it by this method, we say that the exam *is in class* P. For instance, if the exam will consist of verifying various multiplication problems (e.g. "Is $231 * 136 = 31516$?"), then this exam is in class P, since P can put the algorithm for long multiplication, together with a multiplication table, on the index card, and perform these computations during the exam. A more non-trivial example of an exam in class P would be an exam consisting solely of determining whether various large numbers are prime; here P could be guaranteed to ace the test by writing down on his index card the details of the AKS primality test.

Student NP is similar to P, but is substantially less scrupulous; he has bribed the proctor of the exam to supply him with a full solution key, containing not only the answers, but also the worked computations that lead to that answer (when the answer is "true"). The reason he has asked (and paid) for the latter is that he does not fully trust the proctor to give reliable answers, and is terrified of the impact to his grades if he makes a false positive. Thus, if the answer key asserts that the answer to a question is "true", he plans to check the computations given to the proctor himself before putting down "true"; if he cannot follow these computations, and cannot work out the problem himself, he will play it safe and put down "false" instead.

We will say that the exam *is in class* NP if

NP is guaranteed to ace the exam if the information given to him by the proctor is reliable;
NP is guaranteed not to make a false positive, even if the proctor has given him unreliable information.

For instance, imagine an exam consisting of questions such as "Is Fermat's last theorem provable in ten pages or less?". Such an exam is in the class NP, as the student can bribe the proctor to ask for a ten-page proof of FLT, if such exists, and then would check that proof carefully before putting down "True". This way, the student is guaranteed not to make a false positive (which, in this context, would be a severe embarrassment to any reputable mathematician), and will ace the exam if the proctor actually does happen to have all the relevant proofs available.

It is clear that NP is always going to do at least as well as P, since NP always has the option of ignoring whatever the proctor gives him, and copying P's strategy instead. But how much of an advantage does NP have over P? In particular, if we give P a little bit more time (and a somewhat larger index card), could every exam that is in class NP, also be in class P? This, roughly speaking, is the $P = NP$ problem. It is believed that $P \neq NP$, thus there are exams which NP will ace (with reliable information) and will at least not make a false positive (even with unreliable information), but for which P is not guaranteed to ace, even with a little extra time and space.

— 2. Oracles —

Now let's modify the exams a bit by allowing a limited amount of computer equipment in the exam. In addition to the scratch paper, pencil, and index card, every student in the exam is now also given access to a computer A which can perform a carefully limited set of tasks that are intended to assist the student. Examples of tasks permitted by A could include a scientific calculator, a mathematics package such as Matlab or SAGE, or access to Wikipedia or Google. We say that an exam is *in class* P^A if it can be guaranteed to be aced by P if he has access to A, and similarly the exam *is in class* NP^A if it can be guaranteed to be aced by NP if he has access to

A and the information obtained from the proctor was reliable, and if he is at least guaranteed not to make a false positive with access to A if the information from the proctor turned out to be unreliable. Again, it is clear that NP will have the advantage over P, in the sense that every exam in class P^A will also be in class NP^A. (In other words, the proof that $P \subset NP$ *relativises*.) But what about the converse – is every exam in class NP^A, also in class P^A (if we give P a little more time and space, and perhaps also a slightly larger and faster version of A)?

We now give an example of a computer A with the property that $P^A = NP^A$, i.e. that every exam in class NP^A, is also in class P^A. Here, A is an extremely fast computer with reasonable amount of memory and a compiler for a general-purpose programming language, but with no additional capabilities. (More precisely, A should be a PSPACE-complete language, but let me gloss over the precise definition of this term here.)

Suppose that an exam is in class NP^A, thus NP will ace the exam if he can access A and has reliable information, and will not give any false positive if he can access A and has unreliable information. We now claim that P can also ace this exam, if given a little bit more time and a slightly larger version of A. The way he does it is to program his version of A to simulate NP's strategy, by looping through all possible values of the solution key that NP might be given, and also simulating NP's copy of A as well. (The latter task is possible as long as P's version of A is slightly larger and faster than NP's version.) There are of course an extremely large number of combinations of solution key to loop over (for instance, consider how many possible proofs of Fermat's last theorem under ten pages there could be), but we assume that the computer is so fast that it can handle all these combinations without difficulty. If at least one of the possible choices for a solution key causes the simulation of NP to answer "true", then P will answer "true" also; if instead none of the solution keys cause NP to answer "true", then P will answer "false" instead. If the exam is in class NP^A, it is then clear that P will ace the exam.

Now we give an example of a computer B with the property that $P^B \neq NP^B$, i.e. there exists an exam which is in class NP^B, but for which P is not guaranteed to ace even with the assistance of B. The only software loaded on B is a web browser, which can fetch any web page desired after typing in the correct URL. However, rather than being connected to the internet, the browser can only access a local file system of pages. Furthermore, there is no directory or search feature in this file system; the only way to find a page is to type in its URL, and if you can't guess the URL correctly, there is no way to access that page. (In particular, there are no links between pages.)

Furthermore, to make matters worse, the URLs are not designed according to any simple scheme, but have in fact been generated randomly, by the following procedure. For each positive integer n, flip a coin. If the coin is heads, then create a URL of n random characters and place a web page at that URL. Otherwise, if the coin is tails, do nothing. Thus, for each n, there will either be one web page with a URL of length n, or there will be no web pages of this length; but in the former case, the web page will have an address consisting of complete gibberish, and there will be no means to obtain this address other than by guessing.

The exam will consist of a long series of questions such as "Is there a web page on B with a URL of 1254 characters in length?".

It is clear that this exam is in class NP^B. Indeed, for NP to ace this exam, he just needs to bribe the proctor for the URLs of all the relevant web pages (if they exist). He can then confirm their existence by typing them into B, and then answer "true" if he finds the page, and "false" otherwise. It is clear that NP will ace the exam if the proctor information is reliable, and will avoid false positives otherwise.

On the other hand, poor P will have no chance to ace this exam if the length of the URLs are long enough, for two reasons. Firstly, the browser B is useless to him: any URL he can guess will have almost no chance of being the correct one, and so the only thing he can generate on the browser is an endless stream of "404 Not Found" messages. (Indeed, these URLs are very likely to have a high Kolmogorov complexity, and thus cannot be guessed by P. Admittedly, P does have B available, but one can show by induction on the number of queries that B is useless to P. We also make the idealised assumption that side-channel attacks are not available.) As B is useless, the only hope P has is to guess the sequence of coin flips that were used to determine the set of n for which URLs exist of that length. But the random sequence of coin flips is also likely to have high Kolmogorov complexity, and thus cannot be guaranteed to be guessed by P either. Thus $P^B \neq NP^B$.

> **Remark 1** Note how the existence of long random strings could be used to make an oracle that separates P from NP. In the absence of oracles, it appears that separation of P from NP is closely connected to the existence of long *pseudorandom* strings – strings of numbers which can be deterministically generated (perhaps from a given seed) in a reasonable amount of time, but are difficult to distinguish from genuinely random strings by any quick tests. See my writeup of this lecture by Avi Wigderson for more discussion.

SHARE THIS:

Print Email More

★ Like

6 bloggers like this.

36 comments

Comments feed for this article

1 August, 2009 at 9:05 pm
D. Eppstein

it appears that separation of {P} from {NP} is closely connected to the existence of long pseudorandom strings

Huh. If P=NP then P=BPP, because BPP is part of the polynomial hierarchy. Intuitively, the way for P=BPP would be for there to exist a deterministic pseudorandom number generator that cannot be distinguished from truly random within P — then any BPP program could be derandomized using this generator. But, being able to invert a pseudorandom number generator (distinguishing it from random) is an NP task. So if P=BPP then (waves hands extra-vigorously) there exists an NP task that can't be performed within P, namely inverting the supposed pseudorandom number generator, and therefore P≠NP.

In summary: P=NP => P=BPP => P≠NP or more simply P≠NP.

Sadly, I doubt this line of reasoning has much probability of leading to a million-dollar prize for me.

12 11 Rate This

Reply

1 August, 2009 at 10:25 pm
Suresh

This sounds like razborov-rudich in disguise….

4 4 Rate This

Reply

2 August, 2009 at 7:21 am
Björn Edström

Good exposition, thanks. A good follow up would be an anology about natural proofs. I think I'm not the only one who would appreciate such an article.

9 0 Rate This

Reply

4 August, 2009 at 11:24 am

Harrison

The problem with natural proofs is I'm not sure what the proper analogy for circuits is in this metaphor. Personally, I'd like to see a multiple-choice explanation of algebrization, which I feel I understand a lot less and which is more obviously connected with oracles.

1 0 Rate This

Reply

2 August, 2009 at 10:03 am

rasmus

Are you going to work on P vs NP?

10 11 Rate This

Reply

2 August, 2009 at 10:29 am

Andy D

These are useful analogies. As Terry says, the simple ideas of oracle results can be obscured by technical details arising from the TM model. In practice, I think complexity theorists feel the same way, and have developed ways of defining and thinking about oracles that make life easier, although these techniques may not always come thru clearly in papers (and may not have as much narrative quality as Terry's story).

The main idea is usually to view Turing machines as taking oracles as input, and trying to solve problems defined in terms of the oracle's bits on a particular input length. The Turing machine's access to the oracle is quite limited, hopefully to the extent that there are provable limitations on its ability to reason about the oracle.

For instance, the existence of an oracle relative to which P^A != NP^A is often seen as a 'translation' or 'scaling up' of the fact that the OR function on N bits has deterministic query complexity N, but nondeterministic query complexity 1.

Another example: the existence of an oracle relative to which the Polynomial Hierarchy (PH) is properly contained in PSPACE, is seen (and proved) as a translation of the fact that an efficiently computable function (namely, PARITY on N bits) requires exponential-size constant-depth AND/OR circuits (aka 'AC^0 circuits') to compute.

This seems to've actually been the original motivation to study constant-depth circuits. A lot of research in the more 'concrete' models of query- and circuit-based computation received its impetus from the more 'abstract' world of oracles in the Turing machine model. Of course, concrete computational models are also interesting to study in their own right, but they remain useful frameworks thru which to view oracle questions in complexity.

Unfortunately, I'm not aware of any single paper that comprehensively teaches the use of this perspective and how to make the appropriate translations. Maybe someone else can chime in with one.

9 0 Rate This

Reply

2 August, 2009 at 10:45 am

Andy D

What keeps David's sketch-argument from being more lucrative is partly the fact that derandomization hypotheses come in a variety of strengths. There are three that are relevant here, which I'll list in increasing order of strength.

H1: P = BPP.

H2: For every c > 0, there exists a poly-time-computable pseudorandom generator that fools any distinguisher running in time n^c.

H3: There exists a poly-time-computable pseudorandom generator that fools every poly-time distinguisher.

(I’m neglecting the additional variable of the seed-length for the generator; assume this is logarithmic in the output length.)

Now, it’s not known that H1 implies H2 or that H2 implies H3. David suggests that ‘intuitively’ P = BPP would be established by proving H3, which indeed would imply P != NP. And indeed, H3 is quite plausible, and follows from the existence of one-way functions.

However, the reason most complexity theorists now strongly believe P=BPP, is due to papers (roughly speaking, starting with Nisan-Wigderson and culminating in Impagliazzo-Wigderson) that give plausible computational-hardness assumptions which would imply H2 and therefore P=BPP. But neither H2 nor the hardness assumptions used in these papers seem to imply P != NP.

10 0 Rate This
Reply

2 August, 2009 at 11:03 am
Tom

on second line should be “whether A problem” *[Corrected, thanks – T]*.

1 0 Rate This
Reply

2 August, 2009 at 4:06 pm
Random bits « Equilibrium Networks

[…] Terry Tao on why “there cannot be any relativisable proof of either or “ […]

0 6 Rate This
Reply

2 August, 2009 at 6:13 pm
hezhigang

description of Millennium Problems of CMI
http://www.claymath.org/millennium/P_vs_NP/

http://claymath.msri.org/pversusnp.mov
the lecturer is a woman, her explaination is pleasure to watch.

3 10 Rate This
Reply

3 August, 2009 at 4:14 pm
Greg Kuperberg

The metaphor of a student taking a test is extremely useful for understanding many complexity classes and their interactions with oracles. In my opinion, though, you don’t need to be as fancy as to allow computer assistance; you could instead have the oracle be a thick textbook. This is clearer as a metaphor because the textbook is clearly static, and its contents are well-controlled. Oracles are required to be static. By contrast, the rule of an interactive proof system is that the Merlin (or the ringer in your scenario) does not have to answer predictably.

Another example is the oracle that separates BPP from P. In this construction, most pages of the textbook simply state the correct answer. Thus, a student capable of making random choices can probably provide a correct answer simply by opening the book to a random page. However, the textbook is constructed to penalize uncreative students. The nth exam and the nth textbook are designed so that all of the pages visited by the nth deterministic student are blank. That student won’t learn the correct answer from the book. In other words, the textbook is a kind of conspiracy.

An oracle can also be used to address David Eppstein’s question. In this case you just let the allowed book be filled with random numbers, like just the phone numbers from a phone book. On the one hand, it can be shown that such a book works as an effective pseudo-random number generator and renders BPP equal to P. On the other hand, a book with random contents also works for Terry’s construction and separates NP from P. At least the simplest derandomization arguments relativize, so this construction shows you that the assumptions must be stronger than just that P does not equal NP.

4 0 Rate This
Reply

4 August, 2009 at 2:49 pm
Greg Kuperberg

Sorry, the random oracle does not address David Eppstein's question, because relative to that oracle P = BPP as expected and P ≠ NP, also as expected. However, the conspiracy oracle does address the question. Some (or all?) derandomization arguments relativize, yet there is an oracle relative to which P ≠ BPP.

That may sound paradoxical, but it's not, because a derandomization argument can use an extra resource, or it can depend on an assumption that does not hold relative to some oracle. For instance, the first such argument was that BPP is contained in P/poly, where "/poly" is trustworthy advice that depends on the length but not the content of the input. Instead of an untrustworthy ringer who wants you to say "yes" to yes/no questions, the advice might be written on the board by an honest exam proctor. But the advice can only depend on the time allotted for the exam (say) and not on the contents of the exam. (Otherwise the result would be the universal complexity class ALL, because the proctor could write the answers on the board.) The idea of the proof of this inclusion is that the deterministic student should ask the proctor to write a random string on the board. Even though the string is fixed, if it was randomly chosen beforehand, it can serve as a source of pseudo-random numbers.

0 0 Rate This
Reply

3 August, 2009 at 4:26 pm
Top Posts « WordPress.com

[…] P=NP, relativisation, and multiple choice exams The most fundamental unsolved problem in complexity theory is undoubtedly the P=NP problem, which asks (roughly […] […]

0 2 Rate This
Reply

12 August, 2009 at 8:08 am
Tao on relativisation | An Academic Log of Zirui

[…] Read: https://terrytao.wordpress.com/2009/08/01/pnp-relativisation-and-multiple-choice-exams/ […]

0 0 Rate This
Reply

5 November, 2009 at 8:49 am
The "no self-defeating object" argument « What's new

[…] basically because such arguments tend to be relativisable whereas the problem is not; see this earlier blog post for more discussion. On the other hand, one has the curious feature that many proposed proofs that […]

0 0 Rate This
Reply

8 August, 2010 at 7:17 pm
Adam

Prof Tao — did you see this?

http://www.hpl.hp.com/personal/Vinay_Deolalikar/Papers/pnp_preliminary.pdf

3 0 Rate This
Reply

9 August, 2010 at 1:55 am
Terence Tao

This paper is already being discussed at the blogs of Lipton and of Aaronson, both of whom are more expert in this area than I am. Readers who are interested in this paper should probably follow the discussions there.

My initial impressions are that the paper does not raise any of the usual and obvious "red flags" that accompany the majority of proposed solutions to famous problems, but also does not yet align comfortably with the established progress on this problem (in particular, the focus on the properties of random k-SAT seems to be incongruous with what we have learned from the work of Razbarov and Rudich, even if it may technically be compatible), or on related problems in complexity theory.

It may take some time for an understanding of how this paper fits with the existing body of work on the problem to emerge, and this in turn means that it will also take time for experts to perform a careful verification of this rather lengthy manuscript. (The latter is of course needed to ensure that the argument is *correct*, but the former is more important for ensuring that the argument is *convincing*, and also for salvaging useful mathematics out of it in case there turns out to be a gap.) So some patience may be required before we can get a clearer evaluation of the paper; even in this internet age, mathematical research usually does not follow the 24-hour news cycle.

15 0 Rate This

Reply

9 August, 2010 at 1:37 pm
AJ

Regarding compatibility with Razbarov and Rudich: http://arxiv.org/abs/0805.1385 says a modification to the technique which R-R excluded, might still work. This make Vinay's work more possible to be technically compatible with RR work. Although Vinay's proof is non constructive, it might just work.

0 1 Rate This

Reply

9 August, 2010 at 7:59 pm
Terence Tao

Actually, I may need to revise my last comment; there is now the beginnings of an online effort to collectively understand the argument at

http://rjlipton.wordpress.com/2010/08/09/issues-in-the-proof-that-p≠np

and

http://geomblog.blogspot.com/2010/08/on-deolalikar-proof-crowdsourcing.html

So the time scales here may in fact be shorter than in previous events of this type.

Still, even if these efforts find problems with the specific manuscript, the question of what can be salvaged from this paper (which does seem to contain a number of innovative new ideas) will still probably take a while to resolve.

2 1 Rate This

Reply

9 August, 2010 at 9:41 pm
AJ

Just curious. If statistical physics was powerful enough to attack P != NP, why was it not used to give alternate proofs of simpler complexity results?

1 1 Rate This

Reply

10 August, 2010 at 12:00 am
Terence Tao

Perhaps this event is moving at internet speeds after all. There is now a clearinghouse wiki page for analysis of the Deolalikar paper hosted on the Polymath wiki; this is derived from an earlier online document of Suresh Venkatasubramanian. It is now open for corrections and contributions. At this stage, though, it is not officially a polymath project.

2 0 Rate This

Reply

10 August, 2010 at 1:24 pm
AJ

I hope to decide there is a bug in his proof or not is not np-hard:)

4 0 Rate This

18 August, 2010 at 5:17 pm
AJ

Great collaborative work Prof Tao. I really enjoyed many of your insightful comments including the car having 200mpg analogy.

0 0 Rate This

10 August, 2010 at 12:36 am
tushar das

The following is from

http://www.hpl.hp.com/personal/Vinay_Deolalikar/

"Vinay Deolalikar. P is not equal to NP. 6th August, 2010 (66 pages 10pt, 102 pages 12pt). Manuscript sent on 6th August to several leading researchers in various areas. Confirmations began arriving 8th August early morning. The preliminary version made it to the web without my knowledge. I have made minor updates, here. Please note that the final version of the paper is under preparation, and is to be posted here very shortly. Stay tuned. "

0 2 Rate This
Reply

10 August, 2010 at 2:55 am
Vinay Deolalikar says P ≠ NP « viXra log

[…] about ordered and unordered structures. For more details follow the discussions here, here, here, here and […]

0 0 Rate This
Reply

11 August, 2010 at 6:16 am
My pennyworth about Deolalikar « Gowers's Weblog

[…] 6. I am not sure about this one, but I have a feeling that experts would tell me that any attempt along these lines would relativize: that is, it would also prove the false result that PNP relative to any oracle. I don't want to say more than that because my understanding of this barrier is such that I am likely to get things wrong. Instead, let me refer you to this post of Terry Tao. […]

0 1 Rate This
Reply

12 August, 2010 at 2:29 pm
AJ

I saw this comment on the wiki ".....the discrete probabilistic distributions in the paper can be viewed as tensors, or very special multilinear polynomials. The assumptions "P=NP" somehow gives a (polynomial?) upper bound on the tensor rank. And finally, using known probabilistic results, he gets nonmatching (exponential?) lower bound on the same rank. If I am right, then this approach is a very clever, in a good sense elementary, way to push the previous algebraic-geometric approaches"… is there an easier way to understand this statement with an example?
Thank you

0 2 Rate This
Reply

14 August, 2010 at 1:13 pm
Muhammad Alkarouri

On an administrational note, this post does not appear in the blog's front page nor in the RSS feed. Is this deliberate?

0 0 Rate This
Reply

18 August, 2010 at 4:46 am
Peter Drubetskoy

You seemed to not have used diagonalization in coming up with the oracle B (as is usual in the proof of Baker-Gill-Solovay) but rather randomness ("un-guessability"?) of B's URLs and the coin flips. Are these somehow equivalent? Or is diagonalization not essential for the proof?
I still cannot seem to get my head around the idea that showing that two objects (viz. complexity classes) are not equal under some "transformation" is **not** good for showing them unequal. Here the "transformation" is the oracle and it is clearly "not good" since even classes that are one and the same (IP=PSPACE) can be different in the presence of an oracle. Is there an intuitive way you can think of to explain this phenomenon?

0 0 Rate This
Reply

18 August, 2010 at 7:44 am
Terence Tao

In keeping with the analogy of the main post: it is entirely possible to have two students who are equally competent at exams when not given any external assistance, to have differing levels of competence when given a form of assistance (e.g. a cheat sheet).

Diagonalisation and randomisation are both ways to achieve the same purpose, namely to avoid an unwanted collision (or, in the language of the above analogy, to avoid the student ever "getting lucky" and finding the correct URL by accident). The underlying intuition (that finding a (randomly or diagonally selected) needle in an exponentially large haystack should be a very computationally difficult task) is probably more important here than the specific tool used to formalise that intuition.

0 0 Rate This
Reply

18 August, 2010 at 1:17 pm
Peter Drubetskoy

Thanks! I guess you just highlighted the fact that for complexity classes to be "equal" means "accepting the same languages", not really about being "one and the same". I wonder if all such relativazing results work in one direction – i.e., there exists A such that NP^A "is stronger" than P^A, so, in some respect even if P=NP, NP is still somehow "richer". Or do there exist complexity classes X and Y such that, for some oracle A, X^A \subset Y^A and, for some oracle B, Y^B \subset X^B – that is, X and Y both have "potential strengths" relative to each other.

0 0 Rate This
Reply

28 November, 2012 at 1:43 pm
Frank

Dear Tao

Could I have your opinion about this point of view?

http://the-point-of-view-of-frank.blogspot.com/2012/08/p-versus-up.html

0 4 Rate This
Reply

31 December, 2012 at 4:19 pm
Jasper

Dear Dr. Tao,

I think I have an equation solution for the problem, would you like to read it?

Sincerely yours,
Jasper

0 5 Rate This
Reply

1 May, 2015 at 5:59 pm
vznvzn

hey for one would wish youd write more on P vs NP & complexity theory… :|

:arrow: :star: :idea: :!: fyi a quick summary/ overview of some striking new results maybe eventually touching the area & its all just begging for some hardcore mathematicians to jump into the *fray*: arithmetic circuit complexity, Valiants VP=?VNP, recent advances/ breakthroughs, new directions :D :cool:

elsewhere more on P vs NP as a "math monster/ everest/ olympus mons"

0 4 Rate This
Reply

2 November, 2015 at 3:15 am
Anonymous

The P = NP problem is demonstrated.

P = NP, in the factorization of all odd number.

See: http://www.ijma.info Vol 6 (9) 2015

0 4 Rate This
Reply

27 April, 2016 at 12:25 pm
gninrepoli

Just my intuition. I believe that the Riemann hypothesis is connected with concentration of measure – ttps://terrytao.wordpress.com/2010/01/03/254a-notes-1-concentration-of-measure/.In particular, this expression: $\mid \pi(x) - Li(x) \mid < \frac{x}{log(x)}$.Here's another hint that this hypothesis may be associated with random variables - https://en.wikipedia.org/wiki/Riemann_hypothesis#Number_of_zeros, Ghosh proof("…resembles a Gaussian random variable with mean 0 and variance…").If this hypothesis is in fact associated with random variables, we can make that kind of that particular formulation of the problem (relying on RH) is $\mathbf{NP-complete}$.

1 0 Rate This
Reply