hypothesis in terms of primes should be similar, but would assert that the error term in the prime number theorem (as well as variants of this theorem for almost primes) in short intervals enjoys the expected normal distribution; I do not know of a precise formulation of this assertion, but calculations in this direction lie in [**BoKe1996**].

Notes. This article first appeared at

`terrytao.wordpress.com/2009/07/12`.

Thanks to anonymous commenters for corrections.

Emmanuel Kowalski discusses the relationship between Selberg's limit theorem and the Erdős-Kac theorem further at

`http://blogs.ethz.ch/kowalski/2009/02/28/a-beautiful-analogy-2/`.

1.9. $P = NP$, relativisation, and multiple-choice exams

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 *nondeterministic 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 nondeterministic 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 nondeterministic 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 nondeterministic 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 [**BaGiSo1975**]) 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 nondeterministic 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.

1.9.1. 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 mathematics 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 nontrivial 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.

1.9.2. Oracles. Now let us 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 positives if he can access A and has unreliable information. We now claim that P can also ace this exam, if he is 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 in the 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 that 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 cannot 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 1.9.1. 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 *pseudo-random* 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.

Notes. This article first appeared at

`terrytao.wordpress.com/2009/08/01`.

Thanks to Tom for corrections.

There was some discussion on the relationship between $P = NP$ and $P = BPP$. Greg Kuperberg gave some further examples of oracles that shed some light on this:

- Consider as an oracle an extremely large book of randomly generated numbers. This oracle could be used to simulate any probabilistic algorithm, so $P = BPP$ relative to this oracle. On the other hand, if one assigns the task to determine whether a given string of numbers exists in some range in the book, this question is in NP but not in P.
- Another example of an oracle would be an extremely large book, in which most of the pages contained the answer to the problem at hand, but for which the nth page was blank for every natural number n that could be quickly created by any short deterministic algorithm. This type of oracle could be used to create a scenario in which $P \neq BPP$ and $P \neq NP$.
- A third example, this time of an *advice function* rather than an oracle, would be if the proctor wrote a long random string on the board before starting the exam (with the length of the string depending on the length of the exam). This can be used to show the inclusion $BPP \subset P/poly$.

By using written oracles instead of computer oracles, it also became more obvious that the oracles were noninteractive (i.e., subsequent responses by the oracle did not depend on earlier queries).

Notes. This article first appeared at

terrytao.wordpress.com/2009/10/19.

Thanks to Lior, Raj, and anonymous commenters for corrections.

Raj and Ben Wieland noted the close connection to the Kan extension property.

1.15. The "no self-defeating object" argument

A fundamental tool in any mathematician's toolkit is that of *reductio ad absurdum*: showing that a statement X is false by assuming first that X is true, and showing that this leads to a logical contradiction. A particularly pure example of *reductio ad absurdum* occurs when establishing the nonexistence of a hypothetically overpowered object or structure X, by showing that X's powers are "self-defeating": the very existence of X and its powers can be used (by some clever trick) to construct a counterexample to that power. Perhaps the most well-known example of a self-defeating object comes from the *omnipotence paradox* in philosophy: "Can an omnipotent being create a rock so heavy that He cannot lift it?" More generally, a large number of other paradoxes in logic or philosophy can be reinterpreted as a proof that a certain overpowered object or structure does not exist.

In mathematics, perhaps the first example of a self-defeating object one encounters is that of a largest natural number:

Proposition 1.15.1 (No largest natural number)**.** *There does not exist a natural number N which is larger than all other natural numbers.*

Proof. Suppose for contradiction that there was such a largest natural number N. Then $N+1$ is also a natural number which is strictly larger than N, contradicting the hypothesis that N is the largest natural number. □

Note the argument does not apply to the *extended natural number system* in which one adjoins an additional object ∞ beyond the natural numbers, because $\infty + 1$ is defined equal to ∞. However, the above argument does show that the existence of a largest number is not compatible with the *Peano axioms*.

This argument, by the way, is perhaps the only mathematical argument I know of which is routinely taught to primary school children *by other primary school children*, thanks to the schoolyard game of naming the largest number. It is arguably one's first exposure to a mathematical *nonexistence result*, which seems innocuous at first but can be surprisingly deep, as such results preclude in advance all future attempts to establish existence of that object, no matter how much effort or ingenuity is poured into this task.

One sees this in a typical instance of the above schoolyard game; one player tries furiously to cleverly construct some impressively huge number N, but no matter how much effort is expended in doing so, the player is defeated by the simple response "... plus one!" (unless, of course, N is infinite, ill defined, or otherwise not a natural number).

It is not only individual objects (such as natural numbers) which can be self-defeating; structures (such as orderings or enumerations) can also be self-defeating. (In modern set theory, one considers structures to themselves be a kind of object, and so the distinction between the two concepts is often blurred.) Here is one example (related to, but subtly different from, the previous one):

Proposition 1.15.2 (The natural numbers cannot be finitely enumerated)**.** *The natural numbers* $\mathbf{N} = \{0, 1, 2, 3, \ldots\}$ *cannot be written as* $\{a_1, \ldots, a_n\}$ *for any finite collection* $a_1, \ldots, a_n$ *of natural numbers.*

Proof. Suppose for contradiction that such an enumeration $\mathbf{N}=\{a_1, \ldots, a_n\}$ existed. Then consider the number $a_1 + \cdots + a_n + 1$; this is a natural number, but it is larger than (and hence not equal to) any of the natural numbers $a_1, \ldots, a_n$, contradicting the hypothesis that $\mathbf{N}$ is enumerated by $a_1, \ldots, a_n$. □

Here, it is the *enumeration* which is self-defeating, rather than any individual natural number such as a_1 or a_n. (For this article, we allow enumerations to contain repetitions.)

The above argument may seem trivial, but a slight modification of it already gives an important result, namely *Euclid's theorem*:

Proposition 1.15.3 (The primes cannot be finitely enumerated)**.** *The prime numbers* $\mathcal{P} = \{2, 3, 5, 7, \ldots\}$ *cannot be written as* $\{p_1, \ldots, p_n\}$ *for any finite collection of prime numbers.*

Proof. Suppose for contradiction that such an enumeration $\mathcal{P}=\{p_1, \ldots, p_n\}$ existed. Then consider the natural number $p_1 \times \cdots \times p_n + 1$; this is a natural number larger than 1 which is not divisible by any of the primes $p_1, \ldots, p_n$. But, by the *fundamental theorem of arithmetic* (or by the method of *infinite descent*, which is another classic application of *reductio ad absurdum*), every natural number larger than 1 must be divisible by some prime, contradicting the hypothesis that $\mathcal{P}$ is enumerated by $p_1, \ldots, p_n$. □

Remark 1.15.4. Continuing the number-theoretic theme, the "dueling conspiracies" arguments in Section 1.12.4 can also be viewed as an instance of this type of "no self-defeating object"; for instance, a zero of the Riemann zeta function at $1 + it$ implies that the primes anticorrelate almost completely with the multiplicative function n^{it}, which is self-defeating because

it also implies complete anticorrelation with n^{-it}, and the two are incompatible. Thus we see that the *prime number theorem* (a much stronger version of Proposition 1.15.3) also emerges from this type of argument.

In this article I would like to collect several other well-known examples of this type of "no self-defeating object" argument. Each of these is well studied and probably quite familiar to many of you, but I feel that by collecting them all in one place, the commonality of theme between these arguments becomes more apparent. (For instance, as we shall see, many well-known "paradoxes" in logic and philosophy can be interpreted mathematically as rigorous "no self-defeating object" arguments.)

1.15.1. Set theory. Many famous foundational results in set theory come from a "no self-defeating object" argument. (Here, we shall be implicitly using a standard axiomatic framework for set theory, such as *Zermelo-Frankel set theory*; the situation becomes different for other set theories, much as results such as Proposition 1.15.1 change if one uses other number systems such as the extended natural numbers.) The basic idea here is that any sufficiently overpowered set-theoretic object is capable of encoding a version of the *liar paradox* ("this sentence is false", or more generally a statement which can be shown to be logically equivalent to its negation) and thus lead to a contradiction. Consider for instance this variant of *Russell's paradox*:

Proposition 1.15.5 (No universal set)**.** *There does not exist a set which contains all sets (including itself).*

Proof. Suppose for contradiction that there existed a universal set X which contained all sets. Using the *axiom schema of specification*, one can then construct the set

$$Y := \{A \in X : A \notin A\}$$

of all sets in the universe which did not contain themselves. As X is universal, Y is contained in X. But then, by definition of Y, one sees that $Y \in Y$ if and only if $Y \notin Y$, a contradiction. □

Remark 1.15.6. As a corollary, there also does not exist any set Z which contains all *other* sets (not including itself), because the set $X := Z \cup \{Z\}$ would then be universal.

One can "localise" the above argument to a smaller domain than the entire universe, leading to the important

Proposition 1.15.7 (Cantor's theorem)**.** *Let X be a set. Then the power set $2^X := \{A : A \subset X\}$ of X cannot be enumerated by X, i.e., one cannot write $2^X := \{A_x : x \in X\}$ for some collection $(A_x)_{x \in X}$ of subsets of X.*

Proof. Suppose for contradiction that there existed a set X whose power set 2^X could be enumerated as $\{A_x : x \in X\}$ for some $(A_x)_{x \in X}$. Using the axiom schema of specification, one can then construct the set

$$Y := \{x \in X : x \notin A_x\}.$$

The set Y is an element of the power set 2^X. As 2^X is enumerated by $\{A_x : x \in X\}$, we have $Y = A_y$ for some $y \in X$. But then by the definition of Y, one sees that $y \in A_y$ if and only if $y \notin A_y$, a contradiction. □

As is well known, one can adapt Cantor's argument to the real line, showing that the reals are uncountable:

Proposition 1.15.8 (The real numbers cannot be countably enumerated)**.** *The real numbers* $\mathbf{R}$ *cannot be written as* $\{x_n : n \in \mathbf{N}\}$ *for any countable collection* $x_1, x_2, \ldots$ *of real numbers.*

Proof. Suppose for contradiction that there existed a countable enumeration of $\mathbf{R}$ by a sequence $x_1, x_2, \ldots$ of real numbers. Consider the decimal expansion of each of these numbers. Note that, due to the well-known "$0.999\cdots = 1.000\cdots$" issue, the decimal expansion may be nonunique, but each real number x_n has at most two decimal expansions. For each n, let $a_n \in \{0, 1, \ldots, 9\}$ be a digit which is not equal to the nth digit of any of the decimal expansions of x_n; this is always possible because there are ten digits to choose from and at most two decimal expansions of x_n. (One can avoid any implicit invocation of the *axiom of choice* here by setting a_n to be (say) the *least* digit which is not equal to the nth digit of any of the decimal expansions of x_n.) Then the real number given by the decimal expansion $0.a_1a_2a_3\cdots$ differs in the nth digit from any of the decimal expansions of x_n for each n, and so is distinct from every x_n, a contradiction. □

Remark 1.15.9. One can of course deduce Proposition 1.15.8 directly from Proposition 1.15.7 by using the decimal representation to embed $2^{\mathbf{N}}$ into $\mathbf{R}$. But notice how the two arguments have a slightly different (though closely related) basis; the former argument proceeds by encoding the liar paradox, while the second proceeds by exploiting Cantor's diagonal argument. The two perspectives are indeed a little different: for instance, Cantor's diagonal argument can also be modified to establish the *Arzelá-Ascoli theorem*, whereas I do not see any obvious way to prove that theorem by encoding the liar paradox.

Remark 1.15.10. It is an interesting psychological phenomenon that Proposition 1.15.8 is often considered far more unintuitive than any of the other propositions here, despite being in the same family of arguments; most people have no objection to the fact that every effort to finitely enumerate the natural numbers, for instance, is doomed to failure, but for some reason it

is much harder to let go of the belief that, at some point, some sufficiently ingenious person will work out a way to countably enumerate the real numbers. I am not exactly sure why this disparity exists, but I suspect it is at least partly due to the fact that the rigorous construction of the real numbers is quite sophisticated and often not presented properly until the advanced undergraduate level. (Or perhaps it is because we do not play the game "enumerate the real numbers" often enough in schoolyards.)

Remark 1.15.11. One can also use the diagonal argument to show that any reasonable notion of a "constructible real number" must itself be non-constructive (this is related to the *interesting number paradox*). Part of the problem is that the question of determining whether a proposed construction of a real number is actually well defined is a variant of the *halting problem*, which we will discuss below.

While a genuinely universal set is not possible in standard set theory, one at least has the notion of an *ordinal*, which contains all the ordinals less than it. (In the discussion below, we assume familiarity with the theory of ordinals.) One can modify the above arguments concerning sets to give analogous results about the ordinals. For instance:

Proposition 1.15.12 (Ordinals do not form a set)**.** *There does not exist a set that contains all the ordinals.*

Proof. Suppose for contradiction that such a set existed. By the axiom schema of specification, one can then find a set Ω which consists precisely of all the ordinals and nothing else. But then $\Omega \cup \{\Omega\}$ is an ordinal which is not contained in Ω (by the *axiom of foundation*, also known as the *axiom of regularity*), a contradiction. □

Remark 1.15.13. This proposition (together with the theory of ordinals) can be used to give a quick proof of *Zorn's lemma*: see Section 2.4 of *Volume I* for further discussion. Notice the similarity between this argument and the proof of Proposition 1.15.1.

Remark 1.15.14. Once one has Zorn's lemma, one can show that various other classes of mathematical objects do not form sets. Consider for instance the class of all vector spaces. Observe that any chain of (real) vector spaces (ordered by inclusion) has an upper bound (namely the union or limit of these spaces). Thus, if the class of all vector spaces was a set, then Zorn's lemma would imply the existence of a maximal vector space V. But one can simply adjoin an additional element not already in V (e.g., $\{V\}$) to V, and contradict this maximality. (An alternate proof: every object is an element of some vector space, and in particular every set is an element of some vector space. If the class of all vector spaces formed a set, then by the *axiom of*

union, we see that union of all vector spaces is a set also, contradicting Proposition 1.15.5.)

One can localise the above argument to smaller cardinalities, for instance:

Proposition 1.15.15 (Countable ordinals are uncountable)**.** *There does not exist a countable enumeration $\omega_1, \omega_2, \ldots$ of the countable ordinals. (Here we consider finite sets and countably infinite sets to both be countable.)*

Proof. Suppose for contradiction that there exists a countable enumeration $\omega_1, \omega_2, \ldots$ of the countable ordinals. Then the set $\Omega := \bigcup_n \omega_n$ is also a countable ordinal, as is the set $\Omega \cup \{\Omega\}$. But $\Omega \cup \{\Omega\}$ is not equal to any of the ω_n (by the axiom of foundation), a contradiction. □

Remark 1.15.16. One can show the existence of uncountable ordinals (e.g., by considering all the well-orderings of subsets of the natural numbers, up to isomorphism), and then there exists a least uncountable ordinal Ω. By construction, this ordinal consists precisely of all the countable ordinals, but is itself uncountable, much as **N** consists precisely of all the finite natural numbers, but is itself infinite (Proposition 1.15.2). The least uncountable ordinal is notorious, among other things, for providing a host of counterexamples to various intuitively plausible assertions in point set topology, and in particular in showing that the topology of sufficiently uncountable spaces cannot always be adequately explored by countable objects such as sequences.

Remark 1.15.17. The existence of the least uncountable ordinal can explain why one cannot contradict Cantor's theorem on the uncountability of the reals simply by iterating the diagonal argument (or any other algorithm) in an attempt to "exhaust" the reals. From *transfinite induction* we see that the diagonal argument allows one to assign a different real number to each countable ordinal, but this does not establish countability of the reals, because the set of all countable ordinals is itself uncountable. (This is similar to how one cannot contradict Proposition 1.15.5 by iterating the $N \to N+1$ map, as the set of all finite natural numbers is itself infinite.) In any event, even once one reaches the first uncountable ordinal, one may not yet have completely exhausted the reals; for instance, using the diagonal argument given in the proof of Proposition 1.15.8, only the real numbers in the interval $[0,1]$ will ever be enumerated by this procedure. (Also, the question of whether *all* real numbers in $[0,1]$ can be enumerated by the iterated diagonal algorithm requires the *continuum hypothesis*, and even with this hypothesis I am not sure whether the statement is decidable.)

1.15.2. Logic. The "no self-defeating object" argument leads to a number of important nonexistence results in logic. Again, the basic idea is to show

that a sufficiently overpowered logical structure will eventually lead to the existence of a self-contradictory statement, such as the liar paradox. To state examples of this properly, one unfortunately has to invest a fair amount of time in first carefully setting up the language and theory of logic. I will not do so here, and instead use informal English sentences as a proxy for precise logical statements to convey a taste (but not a completely rigorous description) of these logical results here.

The liar paradox itself—the inability to assign a consistent truth value to "this sentence is false"—can be viewed as an argument demonstrating that there is no consistent way to interpret (i.e., assign a truth value to) sentences, when the sentences are (a) allowed to be self-referential, and (b) allowed to invoke the very notion of truth given by this interpretation. One's first impulse is to say that the difficulty here lies more with (a) than with (b), but there is a clever trick, known as *Quining* (or *indirect self-reference*), which allows one to modify the liar paradox to produce a non-self-referential statement to which one still cannot assign a consistent truth value. The idea is to work not with fully formed sentences S, which have a single truth value, but instead with *predicates* S, whose truth value depends on a variable x in some range. For instance, S may be "x is two characters long", and the range of x may be the set of strings (i.e., finite sequences of characters). Then for every string T, the statement $S(T)$ (formed by replacing every appearance of x in S with T) is either true or false. For instance, $S($"ab"$)$ is true, but $S($"abc"$)$ is false. Crucially, predicates are themselves strings, and can thus be fed into themselves as input; for instance, $S(S)$ is false. If however U is the predicate "x is sixty-five characters long", observe that $U(U)$ is true.

Now consider the *Quine predicate* Q given by

"x is a predicate whose range is the set of strings, and $x(x)$ is false"

whose range is the set of strings. Thus, for any string T, $Q(T)$ is the sentence

"T is a predicate whose range is the set of strings, and $T(T)$ is false."

This predicate is defined nonrecursively, but the sentence $Q(Q)$ captures the essence of the liar paradox: it is true if and only if it is false. This shows that there is no consistent way to interpret sentences in which the sentences are allowed to come from predicates, are allowed to use the concept of a string, and also are allowed to use the concept of truth as given by that interpretation.

Note that the proof of Proposition 1.15.5 is basically the set-theoretic analogue of the above argument, with the connection being that one can identify a predicate $T(x)$ with the set $\{x : T(x)\,\text{true}\}$.

By making one small modification to the above argument—replacing the notion of truth with the related notion of provability—one obtains the celebrated *Gödel's (second) incompleteness theorem*:

Theorem 1.15.18 (Gödel's incompleteness theorem, informal statement)**.** *No consistent logical system which has the notion of a string, can provide a proof of its own logical consistency. (Note that a proof can be viewed as a certain type of string.)*

Remark 1.15.19. Because one can encode strings in numerical form (e.g., using the *ASCII code*), it is also true (informally speaking) that no consistent logical system which has the notion of a natural number can provide a proof of its own logical consistency.

Proof (Informal sketch only). Suppose for contradiction that one had a consistent logical system inside of which its consistency could be proven. Now let Q be the predicate given by

x is a predicate whose range is the set of strings, and $x(x)$ is not provable

and whose range is the set of strings. Define the *Gödel sentence* G to be the string $G := Q(Q)$. Then G is logically equivalent to the assertion "G is not provable". Thus, if G were false, then G would be provable, which (by the consistency of the system) implies that G is true, a contradiction; thus, G must be true. Because the system is provably consistent, the above argument can be placed inside the system itself, to *prove* inside that system that G must be true; thus G is provable and G is then false, a contradiction. (It becomes quite necessary to carefully distinguish the notions of truth and provability (both inside a system and externally to that system) in order to get this argument straight!) □

Remark 1.15.20. It is not hard to show that a consistent logical system which can model the standard natural numbers cannot *disprove* its own consistency either (i.e., it cannot establish the statement that one can deduce a contradiction from the axioms in the systems in n steps for some natural number n); thus the consistency of such a system is undecidable within that system. Thus this theorem strengthens the (more well-known) first Gödel incompleteness theory, which asserts the existence of undecidable statements inside a consistent logical system which contains the concept of a string (or a natural number). On the other hand, the incompleteness theorem does not preclude the possibility that the consistency of a weak theory could be proven in a strictly stronger theory (e.g., the consistency of Peano arithmetic is provable in Zermelo-Frankel set theory).

Remark 1.15.21. One can use the incompleteness theorem to establish the undecidability of other overpowered problems. For instance, *Matiyasevich's*

theorem demonstrates that the problem of determining the solvability of a system of Diophantine equations is, in general, undecidable, because one can encode statements such as the consistency of set theory inside such a system.

1.15.3. Computability. One can adapt these arguments in logic to analogous arguments in the theory of computation; the basic idea here is to show that a sufficiently overpowered computer program cannot exist, by feeding the source code for that program into the program itself (or some slight modification thereof) to create a contradiction. As with logic, a properly rigorous formalisation of the theory of computation would require a fair amount of preliminary machinery, for instance to define the concept of Turing machine (or some other universal computer), and so I will once again use informal English sentences as an informal substitute for a precise programming language.

A fundamental "no self-defeating object" argument in the subject, analogous to the other liar paradox type arguments encountered previously, is the *Turing halting theorem*:

Theorem 1.15.22 (Turing halting theorem)**.** *There does not exist a program P which takes a string S as input, and determines in finite time whether S is a program (with no input) that halts in finite time.*

Proof. Suppose for contradiction that such a program P existed. Then one could easily modify P to create a variant program Q, which takes a string S as input, and halts if and only if S is a program (with S itself as input) that does not halt in finite time. Indeed, all Q has to do is call P with the string $S(S)$, defined as the program (with no input) formed by declaring S to be the input string for the program S. If P determines that $S(S)$ does not halt, then Q halts; otherwise, if P determines that $S(S)$ does halt, then Q performs an infinite loop and does not halt. Then observe that $Q(Q)$ will halt if and only if it does not halt, a contradiction. □

Remark 1.15.23. As one can imagine from the proofs, this result is closely related to, but not quite identical with, the Gödel incompleteness theorem. That latter theorem implies that the question of whether a given program halts or not in general is undecidable (consider a program designed to search for proofs of the inconsistency of set theory). By contrast, the halting theorem (roughly speaking) shows that this question is *uncomputable* (i.e., there is no algorithm to decide halting in general) rather than *undecidable* (i.e., there are programs whose halting can neither be proven nor disproven).

On the other hand, the halting theorem can be used to establish the incompleteness theorem. Indeed, suppose that all statements in a suitably strong and consistent logical system were either provable or disprovable.

Then one could build a program that determined whether an input string S, when run as a program, halts in finite time, simply by searching for all proofs or disproofs of the statement "S halts in finite time"; this program is guaranteed to terminate with a correct answer by hypothesis.

Remark 1.15.24. While it is not possible for the halting problem for a given computing language to be computable in that language, it is certainly possible that it is computable in a strictly stronger language. When that is the case, one can then invoke *Newcomb's paradox* to argue that the weaker language does not have unlimited "free will" in some sense.

Remark 1.15.25. In the language of *recursion theory*, the halting theorem asserts that the set of programs that do not halt is not a *decidable set* (or a *recursive set*). In fact, one can make the slightly stronger assertion that the set of programs that do not halt is not even a *semidecidable set* (or a *recursively enumerable set*), i.e., there is no algorithm which takes a program as input and halts in finite time if and only if the input program does not halt. This is because the complementary set of programs that do halt is clearly semidecidable (one simply runs the program until it halts, running forever if it does not), and so if the set of programs that do not halt is also semidecidable, then it is decidable (by running both algorithms in parallel; this observation is a special case of *Post's theorem*).

Remark 1.15.26. One can use the halting theorem to exclude overly general theories for certain types of mathematical objects. For instance, one cannot hope to find an algorithm to determine the existence of smooth solutions to arbitrary nonlinear partial differential equations, because it is possible to simulate a Turing machine using the laws of classical physics, which in turn can be modeled using (a moderately complicated system of) nonlinear PDE. Instead, progress in nonlinear PDE has instead proceeded by focusing on much more specific classes of such PDE (e.g., elliptic PDE, parabolic PDE, hyperbolic PDE, gauge theories, etc.)

One can place the halting theorem in a more "quantitative" form. Call a function $f : \mathbf{N} \to \mathbf{N}$ *computable* if there exists a computer program which, when given a natural number n as input, returns $f(n)$ as output in finite time. Define the *Busy Beaver function* $BB : \mathbf{N} \to \mathbf{N}$ by setting $BB(n)$ to equal the largest output of any program of at most n characters in length (and taking no input), which halts in finite time. Note that there are only finitely many such programs for any given n, so $BB(n)$ is well defined. On the other hand, it is uncomputable, even to upper bound:

Proposition 1.15.27. *There does not exist a computable function f such that one has $BB(n) \leq f(n)$ for all n.*

Proof. Suppose for contradiction that there existed a computable function $f(n)$ such that $BB(n) \leq f(n)$ for all n. We can use this to contradict the halting theorem, as follows. First observe that once the Busy Beaver function can be upper bounded by a computable function, then for any n, the run time of any halting program of length at most n can also be upper bounded by a computable function. This is because if a program of length n halts in finite time, then a trivial modification of that program (of length larger than n, but by a computable factor) is capable of outputting the run time of that program (by keeping track of a suitable "clock" variable, for instance). Applying the upper bound for Busy Beaver to that increased length, one obtains the bound on run time.

Now, to determine whether a given program S halts in finite time or not, one simply simulates (runs) that program for time up to the computable upper bound of the possible running time of S, given by the length of S. If the program has not halted by then, then it never will. This provides a program P obeying the hypotheses of the halting theorem, a contradiction. □

Remark 1.15.28. A variant of the argument shows that $BB(n)$ grows faster than any computable function: thus if f is computable, then $BB(n) > f(n)$ for all sufficiently large n. We leave the proof of this result as an exercise to the reader.

Remark 1.15.29. Sadly, the most important unsolved problem in complexity theory, namely the $P \neq NP$ *problem*, does not seem to be susceptible to the "no self-defeating object" argument, basically because such arguments tend to be *relativisable*, whereas the $P \neq NP$ problem is not; see Section 1.9 for more discussion. On the other hand, one has the curious feature that many proposed *proofs* that $P \neq NP$ appear to be self-defeating; this is most strikingly captured in the celebrated work of Razborov and Rudich [**RaRu1997**], who showed (very roughly speaking) that any sufficiently "natural" proof that $P \neq NP$ could be used to disprove the existence of an object closely related to the belief that $P \neq NP$, namely the existence of pseudo-random number generators. (I am told, though, that diagonalisation arguments can be used to prove some other inclusions or noninclusions in complexity theory that are not subject to the relativisation barrier, though I do not know the details.)

1.15.4. Game theory. Another basic example of the "no self-defeating object" argument arises from game theory, namely the *strategy stealing argument*. Consider for instance a generalised version of naughts and crosses (tic-tac-toe), in which two players take turns placing naughts and crosses on some game board (not necessarily square, and not necessarily two-dimensional), with the naughts player going first, until a certain pattern of all naughts or

all crosses is obtained as a subpattern, with the naughts player winning if the pattern is all naughts, and the crosses player winning if the pattern is all crosses. (If all positions are filled without either pattern occurring, the game is a draw.) We assume that the winning patterns for the cross player are exactly the same as the winning patterns for the naughts player (but with naughts replaced by crosses, of course).

Proposition 1.15.30. *In any generalised version of naughts and crosses, there is no strategy for the second player (i.e., the crosses player) which is guaranteed to ensure victory.*

Proof. Suppose for contradiction that the second player had such a winning strategy W. The first player can then *steal* that strategy by placing a naught arbitrarily on the board, and then pretending to be the second player and using W accordingly. Note that occasionally, the W strategy will compel the naughts player to place a naught on the square that he or she has already occupied, but in such cases the naughts player may simply place the naught somewhere else instead. (It is not possible that the naughts player would run out of places, thus forcing a draw, because this would imply that W could lead to a draw as well, a contradiction.) If we denote this stolen strategy by W', then W' guarantees a win for the naughts player; playing the W' strategy for the naughts player against the W strategy for the crosses player, we obtain a contradiction. □

Remark 1.15.31. The key point here is that in naughts and crosses games, it is possible to play a *harmless move*—a move which gives up the turn of play, but does not actually decrease one's chance of winning. In games such as chess, there does not appear to be any analogue of the harmless move, and so it is not known whether black actually has a strategy guaranteed to win or not in chess, though it is suspected that this is not the case.

Remark 1.15.32. The *Hales-Jewett theorem* shows that for any fixed board length, an n-dimensional game of naughts and crosses is unable to end in a draw if n is sufficiently large. An induction argument shows that for any two-player game that terminates in bounded time in which draws are impossible, one player must have a guaranteed winning strategy; by the above proposition, this strategy must be a win for the naughts player. Note, however, that Proposition 1.15.30 provides no information as to *how* to locate this winning strategy, other than that this strategy belongs to the naughts player. Nevertheless, this gives a second example in which the "no self-defeating object" argument can be used to ensure the *existence* of some object, rather than the *nonexistence* of an object. (The first example was the prime number theorem, discussed earlier.)

The strategy-stealing argument can be applied to real-world economics and finance, though as with any other application of mathematics to the real world, one has to be careful as to the implicit assumptions one is making about reality and how it conforms to one's mathematical model when doing so. For instance, one can argue that in any market or other economic system in which the net amount of money is approximately constant, it is not possible to locate a universal trading strategy which is guaranteed to make money for the user of that strategy, since if everyone applied that strategy, then the net amount of money in the system would increase, a contradiction. Note however that there are many loopholes here; it may be that the strategy is difficult to copy or relies on exploiting some other group of participants who are unaware or unable to use the strategy and would then lose money (though in such a case, the strategy is not truly universal as it would stop working once enough people used it). Unfortunately, there can be strong psychological factors that can cause people to override the conclusions of such strategy-stealing arguments with their own rationalisations, as can be seen, for instance, in the perennial popularity of pyramid schemes, or to a lesser extent, market bubbles (though one has to be careful about applying the strategy-stealing argument in the latter case, since it is possible to have net wealth creation through external factors such as advances in technology).

Note also that the strategy-stealing argument also limits the universal predictive power of *technical analysis* to provide predictions other than that the prices obey a *martingale*, though again there are loopholes in the case of markets that are illiquid or highly volatile.

1.15.5. Physics. In a similar vein, one can try to adapt the "no self-defeating object" argument from mathematics to physics, but again one has to be much more careful with various physical and metaphysical assumptions that may be implicit in one's argument. For instance, one can argue that under the laws of special relativity, it is not possible to construct a totally immovable object. The argument would be that if one could construct an immovable object O in one inertial reference frame, then by the *principle of relativity* it should be possible to construct an object O' which is immovable in another inertial reference frame which is moving with respect to the first; setting the two on a collision course, we obtain the classic contradiction between an irresistible force and an immovable object. Note however that there are several loopholes here which allow one to avoid contradiction; for instance, the two objects O, O' could simply pass through each other without interacting.

In a somewhat similar vein, using the laws of special relativity, one can argue that it is not possible to systematically generate and detect *tachyon*

particles (particles traveling faster than the speed of light) because these could be used to transmit localised information faster than the speed of light and then (by the principle of relativity) to send localised information back into the past from one location to a distant one. Setting up a second tachyon beam to reflect this information back to the original location, one could then send localised information back to one's own past (rather than to the past of an observer at a distant location), allowing one to set up a classic *grandfather paradox*. However, as before, there are a large number of loopholes in this argument which could let one avoid contradiction; for instance, if the apparatus needed to set up the tachyon beam is larger than the distance the beam travels (as is for instance the case in *Mexican wave*-type tachyon beams), then there is no causality paradox. Another loophole occurs if the tachyon beam is not fully localised but propagates in space-time in a manner to interfere with the second tachyon beam. A third loophole occurs if the universe exhibits quantum behaviour (in particular, the ability to exist in entangled states) instead of nonquantum behaviour, which allows for such superluminal mechanisms as wave function collapse to occur without any threat to causality or the principle of relativity. A fourth loophole occurs if the effects of relativistic gravity (i.e., general relativity) become significant. Nevertheless, the paradoxical effect of time travel is so strong that this physical argument is a fairly convincing way to rule out many commonly imagined types of faster-than-light travel or communication (and we have a number of other arguments also that exclude more modes of faster-than-light behaviour, though this is an entire article topic in its own right).

Notes. This article first appeared at

`terrytao.wordpress.com/2009/10/27`.

Thanks to Seva Lev and an anonymous commenter for corrections.

1.16. From Bose-Einstein condensates to the nonlinear Schrödinger equation

The *Schrödinger equation*

$$i\hbar\partial_t|\psi\rangle = H|\psi\rangle$$

is the fundamental equation of motion for (nonrelativistic) quantum mechanics, modeling both one-particle systems and N-particle systems for $N > 1$. Remarkably, despite being a *linear* equation, solutions $|\psi\rangle$ to this equation can be governed by a *nonlinear* equation in the large particle limit $N \to \infty$. In particular, when modeling a *Bose-Einstein condensate* with a suitably