A Century of Controversy Over the Foundations of Mathematics

Computer Technology as a Byproduct

GREGORY J. CHAITIN

An edited transcription of Gregory J. Chaitin's 2 March 2000 Carnegie Mellon University School of Computer Science Distinguished Lecture. The speaker was introduced by Manuel Blum. The lecture was videotaped; this is an edited transcript. Chaitin is at the IBM T.J. Watson Research Center in New York.

Thanks very much, Manuel! It's a great pleasure to be here!

We're in a state of euphoria now in the computer business because things are going so well: the Web, e-commerce. It's all paying for our salaries, and it's a nice moment to be around, when things are going so well. But I'd like to make the outrageous claim, that has a little bit of truth, that actually all of this that's happening now with the computer taking over the world, the digitalization of our society, of information in human society, you could say in a way is the result of a philosophical question that was raised by David Hilbert at the beginning of the century.

It's not a complete lie to say that Turing invented the computer in order to shed light on a philosophical question about the foundations of mathematics that was asked by Hilbert. And in a funny way that led to the creation of the computer business.

But I'd like to make the outrageous claim, that has a little bit of truth, that actually all of this that's happening now with the computer taking over the world, the digitalization of our society, of information in human society, you could say in a way is the result of a philosophical question that was raised by David Hilbert at the beginning of the century.

It's not completely true, but there is some truth in it. You know, most historical statements are a lie, so this one isn't that much worse than most others!

So I'd like to explain the philosophical history of the computer. In a way what happened, and I'll tell you more, is that Hilbert said we should formalize all of mathematics, mathematical reasoning. And this failed: It took Gödel and Turing to show that it couldn't be done. It failed in that precise technical sense. But in fact it succeeded magnificently because, formalization of reasoning no, but formalization of algorithms has been the great technological success of our time—computer programming languages!

The original version was previously published in Finite Versus Infinite: Contributions to an Eternal Dilemma, *Calude, C. S.; Paun, G. (eds.); Springer-Verlag, London, 2000, pp. 75–100.*

So if you look back at the history of the beginning of this century, you'll see papers by logicians studying the foundations of mathematics in which they had programming languages. Now you look back and you say this is clearly a programming language! If you look at Turing's paper, of course, there's a machine language. If you look at papers by Alonzo Church, you see the lambda calculus, which is a functional programming language. If you look at Gödel's original paper, you see what to me looks like LISP—it's very close to LISP; the paper begs to be rewritten in LISP!

So I'd like to give you this hidden philosophical history of computer technology, which is how philosophically minded mathematicians set out to solve once and for all the foundational problems of mathematics and did not succeed but helped to create computer technology as a by-product. This was the failure of this project! We're all benefiting from the glorious failure of this project!

However, this project has not completely died—I'm going to start more systematically from the beginning; but I'm trying to give an introduction. It's popular to think, well Gödel did this wonderful thing in 1931 and Turing added a lot of profound stuff in 1936, but the world has moved on from that point. And what I'd like to do is to tell you that in fact I've done some more work in this area.

You may think it's misguided! Most of the world has shrugged and gone on. We had this disappointment. What Gödel and Turing showed is that axiomatic formal reasoning has certain limitations. You can't formalize it all. And at first people were tremendously shocked and then they shrugged and said, so what? Mathematicians went on, ignoring this. And my misfortune or fortune was that I didn't want to shrug. I said, I want to understand this better. And I'm going to tell you the story of my attempt to understand Gödel incompleteness. It's a psychological problem that a good psychiatrist could have cured me of, and then I wouldn't have done any of this work!

So let me start at the beginning and tell you this story of a hundred years of intense worry, crisis, self-doubt, self-examination, and angst about the philosophy of mathematics.

There've been lots of crises in the history of mathematics. Mathematics is not placid, static, and eternal.

One of the first crises was the Pythagorean result that the square root of two is irrational. And the fact that this was a crisis survives in the word "irrational." Remember the Greeks thought that rationality was the supreme goal—Plato! Reason! If a number is called irrational, that means that this was the Gödel incompleteness theorem of ancient Greece. So there was a crisis there.

Another crisis was caused by the calculus. A lot of people said this is nonsense, we're talking about infinitesimals, what is this? Bishop Berkeley was a theologian, and he said pure mathematicians make as little sense as theologians; you can't reject us saying we're unreasonable. The way you deal with evanescent quantities in the calculus—this was before the calculus had a rigorous foundation—is as bad as our theological discussions! So at that time it was pretty bad!

Then there was a crisis about the parallel postulate, about non-Euclidean geometries.

So mathematics is not static and eternal!

But the particular crisis that I want to tell you about goes back a little more than a hundred years to work of Cantor on set theory.

CANTOR

Set theory

My talk is very impractical. We all know that you can have a start-up and in one year make a million dollars if you're lucky with the Web. So this is about how not to make any money with the Web. This is about how to ruin your career by thinking about philosophy instead.

Cantor was obsessed with the notion of infinite, and it's not mentioned that he was obsessed with infinite because he was interested in theology and God, which is edited out from the accounts now, but that was the original idea.

Cantor had the idea that if you have 1, 2, 3, . . . , why stop there?

$$1, 2, 3, \ldots, \omega$$

I'm giving you a cartoon version of Cantor's theory of infinite sets. You put an omega, ω, this is a Greek letter, the lower case of the last letter in the Greek alphabet, that's the reason to pick it. So you just say, I'm going to put another number here instead of stopping with 1, 2, 3, ... This is going to be the first number after all the finite numbers. This is the first transfinite number.

You can keep going for a while.

$$1, 2, 3, \ldots \omega, \omega+1, \omega+2, \ldots$$

And then you have another thing like a copy of 1, 2, 3, . . . : $\omega+1, \omega+2, \omega+3, \ldots$. These are names. And then you say, why stop here? I'm going to put something after all this, so 2ω, $2\omega+1$, $2\omega+2$, $2\omega+3$, and then later 3ω, 4ω . . . . Well, what comes after all of those?

Why stop there? So, ω^2, obviously.

$$1, 2, 3, \ldots \omega, \omega+1, \omega+2, \ldots, 2\omega, 3\omega, 4\omega, \omega^2$$

Then you keep going: $5\omega^2 + 8\omega + 96$! And then much later you get to ω cubed! And then eventually ω to the fourth. You keep going and why stop there? This sequence goes on forever, but let's put something after all of those. So what would that be? That would be obviously ω to the ω. This is starting to get interesting! Then you keep going and you have ω to the ω to the ω.

This is a pretty far-out number already!

$$1, 2, 3, \ldots \omega, \omega+1, \omega+2, \ldots 2\omega, 3\omega, 4\omega, \omega^2, \omega^3, \omega^4, \omega^\omega, \omega^{\omega^\omega}$$

You can see why this is becoming theological. This is the mathematical equivalent of drug addiction. Instead of getting high on alcohol or grass you get high on ideas like this. After a while you don't know where you're standing or what's going on!

Then the next number is ω to the ω to the ω forever.

$$\omega^{\omega^{\omega^{\omega^{\omega^{\cdots}}}}}$$

This number is the smallest solution of the equation

$$x = \omega^x$$

And it's called ε_0, epsilon nought, I don't know why. Because you start having problems with how to name things, because up to here I was using normal algebraic notation just throwing in ω.

So anyway you can see this is fantastic stuff! I don't know whether it's mathematics, but it's very imaginative, it's very pretty, and actually there was a lot of practical spin-off for pure mathematicians from what Cantor was doing.

Some people regarded set theory as a disease. Poincaré, the great French mathematician, said set theory is a disease from which he hoped future generations will recover. But other people redid all of mathematics using the set-theoretic approach. So modern topology and a lot of abstract mathematics of the twentieth century is a result of this more abstract set-theoretic approach, which generalized questions. The mathematics of the nineteenth century was at a lower level in some ways, looking more at special cases and formulas. The mathematics of the twentieth century—it's hard to write a history of mathematics from the year ten thousand looking back because we're right here—but the mathematics of the twentieth century you could almost say is set-theoretical—"structural"' would be a way to describe it. The mathematics of the nineteenth century was concerned with formulas, infinite Taylor series perhaps. But the mathematics of the twentieth century went on to a set-theoretic level of abstraction.

And in part that's due to Cantor, and some people hate it, saying that Cantor wrecked and ruined mathematics by taking it from being concrete and making it wishy-washy, for example, from hard analysis to abstract analysis. Other people loved this.

It was very controversial. And what didn't help is in fact that there got to be some contradictions. It became more than just a matter of opinion. There were some cases in which you got into really bad trouble, you got obvious nonsense out. And the place you get obvious nonsense out in fact is a theorem of Cantor's that says that for any infinite set there's a larger infinite set that is the set of all its subsets, which sounds pretty reasonable. This is Cantor's diagonal argument—I don't have time to give you the details.

So then the problem is that if you believe that for any infinite set there's a set that's even larger, what happens if you apply this to the universal set, the set of everything? The problem is that by definition the set of everything has everything, and this method supposedly would give you a larger set, which is the set of all subsets of everything. So there's got to be a problem, and the problem was noticed by Bertrand Russell.

BERTRAND RUSSELL

Cantor I think may have noticed it, but Bertrand Russell went around telling everyone about it, giving the bad news to everyone! At least Gödel attributes to Russell the recognition that there was a serious crisis.

The disaster that Russell noticed in this proof of Cantor's was the set of all sets that are not members of themselves—that turns out to be the key step in the proof. And the set of all sets that aren't members of themselves sounds like a reasonable way to define a set, but if you ask if it's inside itself or not, whatever you assume you get the opposite; it's a contradiction. It's like saying this statement is false: The set of all sets that are not members of themselves is contained in itself if and only if it's not contained in itself.

So does this mean that some ways of defining sets are bad, or that the universal set gets you into trouble? What's wrong with the set of everything? So it became increasingly clear there was a problem with set theory. I think Russell helped to make it be recognized by everybody that we had a serious crisis and that methods of reasoning that seemed at first sight perfectly legitimate in some cases led to obvious disaster, to contradictions. There were a whole bunch of paradoxes that Russell advertised: the Berry paradox; the one I just mentioned, called the Russell paradox; and the Burali–Forti paradox.

A lot of these paradoxes in fact were really brought to the attention of the world by Russell. Russell would typically have a footnote saying this paradox occurred to me while I was reading a paper by Burali–Forti, so everyone calls it the Burali–Forti paradox. Burali–Forti I think spent his whole life trying to live down this attribution because he didn't believe that mathematics was in trouble!

Okay so there was a crisis, and I think Russell was one of the key figures in this. At this point David Hilbert comes to the rescue.

DAVID HILBERT

David Hilbert was a very important mathematician around the turn of the century. Unlike Poincaré, a very important French mathematician—Hilbert was a very important German mathematician—Hilbert liked set theory. He liked this abstract Cantorian approach. And Hilbert had the idea of solving once and for all these problems. How was he going to do it?

The way Hilbert was going to do it was using the axiomatic method, which of course goes back to Euclid—Hilbert didn't invent this. But he went one, I think, significant step further.

Hilbert's Formal Axiomatic Method

Hilbert said let's use all the technology from symbolic logic, which a lot of people were involved in inventing, and let's go to some final extreme. Because one of the reasons you got into trouble and got contradictions in mathematics with set theory is because words are

very vague. What we want to do to get rid of all these problems in mathematics and in reasoning is to get rid of pronouns; for example, you don't know what pronouns refer to. And there are all kinds of things that are vague in normal language.

Hilbert said that the way to get rid of all these problems is to come up with a finite set of axioms and an artificial language for doing mathematics—this is the idea of formalism taken to the limit.

FORMALISM

Take formalism to the absolute limit and invent a completely artificial language with completely precise rules of the game—artificial grammar and everything—and eliminate all these problems, like the problems that Russell had. This was an ambitious program to once and for all put mathematics on a firm footing.

And one thing that Hilbert emphasized, which was as far as I know a key contribution that he made, was to say that he wanted the rules of the game for this formal axiomatic system for all of mathematics to be so precise that you have a mechanical proof checker. So it's completely certain and objective and mechanical whether a proof obeys the rules or not. There should be no human element; there should be no subjective element; there should be no question of interpretation. If somebody claims they have a proof, it should be absolutely clear, mechanical: to check, does it obey the rules and you proved the theorem, or does it have a mistake, and it fails.

So this is the idea that mathematics should be absolutely black or white; precise, absolute truth. This is the traditional notion of mathematics.

BLACK OR WHITE

The real world we know is an absolute mess—right?—everything's complicated and messy. But the one place where things should be absolutely clear, black or white, is in pure mathematics.

So this is sort of what Hilbert is saying, and he proposed this as a goal, to have this formalization of all of mathematics and eliminate all the problems. Now this was a program; this was not supposed to be something you did over a weekend. Hilbert proposed this as a goal for putting mathematics on a very firm foundation. And he and a group of very bright collaborators, including John von Neumann, set to work on this, and for a while, for thirty years, it looked sort of encouraging. And then—this is a quick summary of a century of work—then as I'm sure all of you know there were a few little problems!

The problems were, in 1931, Kurt Gödel, and in 1936, Alan Turing.

1931, Gödel and 1936, Turing

They showed that it could not be done, that there were fundamental obstacles to formalizing all of mathematics and making mathematics absolutely black and white and absolutely crystal clear. Remember what Hilbert is proposing is that we should formalize all of mathematics so that everyone on Planet Earth can agree that a proof is either correct or incorrect. The rules of the game should be absolutely explicit: It should be an artificial language and then mathematics will give you absolute truth. "Absolute truth" should be underlined in a very beautiful font, and you should hear the angels singing when you say these words! This was the thought—that we mathematicians have absolute truth. It's ours—no one else has it, only us! That was the idea.

The rules of the game should be absolutely explicit: it should be an artificial language and then mathematics will give you absolute truth. "Absolute truth" should be underlined in a very beautiful font, and you should hear the angels singing when you say these words!

So it turns out this doesn't quite work. Why doesn't it work?

Gödel shocked people quite a bit by showing that it couldn't work. It was very, very surprising when Gödel did this in 1931. And Turing went I think more deeply into it. So let me give you a cartoon five-minute summary, my take on what they did.

Gödel starts with "this statement is false," what I'm now saying is a lie, I'm lying. If I'm lying, and it's a lie that I'm lying, then I'm telling the truth! So "this statement is false" is false if and only if it's true—so there's a problem. Gödel considered instead "this statement is unprovable."

"This stmt is unprovable!"

Here unprovable means unprovable from the axioms of Hilbert's formal axiomatic system, unprovable within the system that Hilbert was trying to create.

Now think about a statement that says that it's unprovable. There are two possibilities: It's provable or it's unprovable. This is assuming you can make a statement say it's unprovable, that there's some way to say this within Hilbert's system. That required enormous cleverness: Gödel numbering, trickery for a statement to refer to itself indirectly, because pronouns that say "this" or "I" aren't usually found in mathematical formulas. So this required a lot of cleverness on Gödel's part. But the basic idea is "this statement is unprovable."

So there are two possibilities. Either it's provable or it's unprovable. And this means provable or unprovable from the system that Hilbert had proposed, the final goal of formalizing all of mathematics.

Well, if it's provable, and it says it's unprovable, we're proving something that's false. So that's not very nice. And if it's unprovable and it says it's unprovable, well then it's true what it's saying, that it's unprovable, and we have a hole. Instead of proving something false, we have incompleteness: we have a true statement that our formalization has not succeeded in capturing.

So the idea is that either we're proving false statements, which is terrifying, or we get something, which is not as bad, but is still awful, which is that our formal axiomatic system is incomplete—there's something that's true but we can't prove it within our system. And therefore the goal of formalizing once

and for all all of mathematics ends up on the floor!

Now I don't think that Hilbert really wanted us to formalize all of mathematics. He didn't say that we should all work in an artificial language and have formal proofs. Formal proofs tend to be very long and inhuman and hard to read. I think Hilbert's goal was philosophical. If you believe that mathematics gives absolute truth, then it seems to me that Hilbert has got to be right, that there ought to have been a way to formalize, once and for all, all of mathematics. That's sort of what mathematical logic was trying to do, that's sort of what the axiomatic method was trying to do, the idea of breaking proofs into smaller and smaller steps. Leibniz thought about this, and Boole thought about this, and Frege and Peano and Russell and Whitehead thought about this. It's the idea of making very clear how mathematics operates step by step. So that doesn't sound bad. Unfortunately, it crashes at this point!

So the idea is that either we're proving false statements, which is terrifying, or we get something, which is not as bad, but is still awful, which is that our formal axiomatic system is incomplete—there's something that's true but we can't prove it within our system. And therefore the goal of formalizing once and for all all of mathematics ends up on the floor!

So everyone is in a terrible state of shock at this point. You read essays by Hermann Weyl or John von Neumann saying things like this: I became a mathematician because this was my religion, I believed in absolute truth, here was beauty, the real world was awful, but I took refuge in number theory. And all of a sudden Gödel comes and ruins everything, and I want to kill myself!

So this was pretty awful. However, this

"This stmt is unprovable!"

is a very strange looking statement. And there are ways of rationalizing—human beings are good at that, we don't want to face unpleasant reality. And this unpleasant reality is very easy to shrug off: You just say, well, who cares! The statements I work with normally in mathematics, they're not statements of this kind. This is nonsense! If you do this kind of stupidity, obviously you're going to get into trouble.

But that's rationalizing too far. Because in fact Gödel made this

"This stmt is unprovable!"

into a statement in elementary number theory. In its original form, sure, it's nonsense; who ever heard of a statement in mathematics that says it's unprovable? But in fact Gödel made this into a numerical statement in elementary number theory, in arithmetic. It was a large statement, but in some clever way, involving Gödel numbering of all arithmetic statements using prime numbers, he was writing it so that it looked like a statement in real mathematics. But it really indirectly was referring to itself and saying that it's unprovable.

So that's why there's a problem. But people didn't really know what to make of this. So I would put "surprising" here, surprising, a terrible shock!

1931 Gödel "This stmt is unprovable!" Surprising

Now my reaction as a child reading this proof is that I follow it step by step, but I don't like it. It doesn't appeal to me! Which is good, because if I had said I like it, it's wonderful, finished, I go ahead and become a molecular biologist and start up a biotech company, and now I'd be rich, but I wouldn't have done any work in this area!

Then comes Turing.

1936 Turing

Now I prefer Turing's approach. Turing goes more deeply into this. Turing starts talking about computers. This is the point where it happens!

1936 Turing Computer

Turing has to invent the computer because Hilbert says that there should be a mechanical procedure to decide if a proof is correct or not. Turing says that what Hilbert really means is there should be a computer program for checking proofs. But first Turing has to say what a computer is, it's a Turing machine, and all of this is in a paper of Turing's in 1936, when there were no computers, so it's a fantastic piece of work. And I would like to claim that this is the invention of the computer. These were general-purpose computers, that was the idea, on paper.

What Turing shows is in fact that there is a relatively concrete statement that escapes the power of mathematics. We now think of computers as physical devices, so they're almost like something in physics. It's a machine working away, it's an idealization of that, you have this machine working, and Turing discovers the halting problem.

1936 Turing Computer Halting Problem

The halting problem says there's no way to decide if a computer program will eventually halt.

Now obviously to decide if a computer program halts is the easiest thing in the world. You run it and when you run out of patience, that's it; it doesn't halt as far as you're concerned. Who cares, you can't wait any longer! But what Turing showed is that there's a problem if you put no time limit. This is very abstract mathematics—in the real world there's always a time limit! You can't run a program a million years, a billion years, $10^{10^{10}}$ years! If you put a time limit, the halting problem is very easy to decide, in principle: You just run the program that long and you see, does it halt by that point or not?

But what Turing showed is that if you put no time limit, then there is no solution. There's no way to decide in advance whether a computer program will halt or not. If it halts, you can eventually discover that by running it. The problem is to realize that you've got to give up. So there's no mechanical procedure that will decide in advance if a

computer program will halt or not, and therefore, it turns out, there is no set of mathematical axioms in Hilbert's sense that can enable you to prove whether a program will halt or not.

Because if you could always prove whether a program will halt or not, you could run through all possible proofs in size order and check whether they're correct, and eventually either find a proof the program's going to halt or find a proof it's not going to halt. And this would give you a way to decide in advance whether a program's going to halt.

Now in practice running through all possible proofs requires an astronomical amount of time. Imagine how many proofs are there that are one page long! You'd never get through them! But in principle you can run through all possible proofs in size order and check whether they obey the rules, if it's a Hilbert formal axiomatic system. So if you had a formal axiomatization of mathematics that enabled you to always prove whether a program halts or not, that would give you a mechanical procedure, by running through all possible proofs in size order, to decide whether a program will halt or not. And Turing showed that you can't do it. His proof, by the way, involves Cantor's diagonal argument—all these ideas are connected, but there's no time to go into that.

So I think that Turing's work makes the limits of mathematics seem much more natural because we're talking about a question about a physical device—it's a computer.

1936 Turing Computer Halting Problem Natural

You fantasize a little bit, you make it a theoretical computer, a computer that can go on forever, that never breaks down, that has as much storage as it wants, so that if numbers get too big it can keep going anyway. But that's not too much of a fantasy; we have devices like that everywhere, right? So it sounds much more concrete. The limits of mathematics discovered by Turing sound more serious, more dangerous than the ones that Gödel found.

And this is the invention of the computer, for this crazy kind of theoretical argument! You don't see billions and billions of dollars of technology in this 1936 paper, but it was all there in embryonic form, as von Neumann kept emphasizing: The universal Turing machine is really the notion of a general-purpose programmable computer. You had machines that did calculations before, but they did special-purpose calculations: They were adding machines, mechanical calculating machines, and I used them when I was a kid. But the notion of a computer is Turing's notion of a machine that can do what any calculating machine can do, and that's the idea of software: It's a very general-purpose machine, it's a flexible machine. So it's really there, von Neumann kept saying, very clearly in Turing's paper. So you have this whole technology there!

And in fact Gödel's paper, as I said, uses LISP; there's a programming language hidden in it. In Turing's paper there's a programming language, given explicitly, Turing machines, and it's a machine language. It's actually a very bad machine language; it's a machine that no person in their sane mind would want to program. But Turing wanted to keep it as simple as possible. Obviously, if his paper had included a manual for the machine language of a real machine, it would have been hopeless; no one would have understood it.

Okay, now what happens with all of this? What happens with all of this is that Hilbert dies, World War II comes, and when I'm a child in the 1950s, I could still read essays by John von Neumann talking about all of this, but the world was clearly going in a less philosophical direction. Things were going downhill rapidly until we're all billionaires with our Web start-ups! People were less concerned about philosophy, and computers were becoming a technology, and Turing was very involved in that, and so was von Neumann.

But stupidly I wanted to understand what was going on in the foundations of mathematics, so in a way I'm stuck in the 1930s. I never got past that stage. What happened? What happened with me is that I couldn't accept the fact that everybody said, who cares! Now it's true that there are a lot of things in life besides the foundations of mathematics and epistemology! There're things like having a family, earning a living, wars, politics, lots of stuff out there, obviously! But what I couldn't accept was that even in the world of pure mathematics, mathematicians were saying, so what, in practice we should do mathematics exactly the same as we've always done it; this does not apply to the problems I care about! That was basically the reaction to Gödel's and Turing's work on incompleteness.

At first there was terrible shock; then it went from one extreme to another. Who cares, people would say, it's obvious, or it's irrelevant! This has no impact in practice on how we should do mathematics. I was very unhappy with that. I was obsessed by incompleteness, and I had an idea.

When I was a kid I really wanted to be a physicist, and a lot of mathematicians say I never made it into mathematics really—I never succeeded; I'm still stuck! I wanted to be a physicist, and I got corrupted by a lot of ideas from physics. While all of this crisis was going on in mathematics, there was a parallel crisis going on in physics, which actually started in the 1920s: That's quantum mechanics, and the key date is 1924.

1924—Quantum Mechanics

And that's the whole question of uncertainty and randomness in fundamental physics. So when I was a kid, besides reading essays talking about Gödel's incompleteness theorem saying, "Oh, my God," there were also essays asking what happened to determinism in physics, what happened to predictability, can there be randomness, does God play dice? Einstein said no, God doesn't play dice. He hated quantum mechanics. And everybody else said yes, God plays dice.

God Plays Dice!

Quantum mechanics is the most successful physical theory ever. We get transistors and computers from it. But

even though Einstein helped to contribute to the creation of quantum mechanics, he hated it. So it looks like Einstein was wrong. God does play dice!

So I had a crazy idea. I thought that maybe the problem is larger and Gödel and Turing were just the tip of the iceberg. Maybe things are much worse and what we really have here in pure mathematics is randomness. In other words, maybe sometimes the reason you can't prove something is not because you're stupid or you haven't worked on it long enough; the reason you can't prove something is because there's nothing there! Sometimes the reason you can't solve a mathematical problem isn't because you're not smart enough or you're not determined enough; it's because there is no solution because maybe the mathematical question has no structure, maybe the answer has no pattern, maybe there is no order or structure that you can try to understand in the world of pure mathematics. Maybe sometimes the reason that you don't see a pattern or structure is because there is no pattern or structure!

And one of my motivations was the prime numbers. There's some work on the prime numbers that says that in some ways the prime numbers can be looked at statistically. There seems to be a certain amount of randomness in the distribution of the primes. That's one of the ways that people try to think about the prime numbers. And this even happens in number theory, which is the queen of pure mathematics!

So on the one hand I heard this talk about probabilistic ways of thinking about the primes—this was heuristic—and this stuff about God plays dice in fundamental physics—what goes on in the atom is random—and I begin to think, well, maybe that's what's going on in the foundations of mathematics. This is what I set out to do, and this project took a long time. One of the first steps is clarifying what is meant by randomness. What is meant by lack of structure, lack of order, lack of pattern?

Randomness: Lack of Structure

So this is a kind of a logical notion of randomness rather than a statistical notion of randomness. It's not like in physics where you ask if a physical process like coin tossing is random. I don't care where something comes from. I just look at something and ask if it has structure or pattern or not. So this is logical or structural randomness as opposed to physical unpredictability and randomness. It's different—it's very closely related, but it's different.

And the idea that I came up with—and Kolmogorov came up with at the same time independently—is the idea that something is random if it can't be compressed into a shorter description, if essentially you just have to write it out as it is. In other words, there's no concise theory that produces it. For example, a set of physical data would be random if the only way to publish it is as is in a table, but if there's a theory, you're compressing a lot of observations into a small number of physical principles or laws. And the more the compression, the better the theory: In accord with Occam's razor, the best theory is the simplest theory. I would say that a theory is a program—also Ray Solomonoff did some thinking along these lines for doing induction—he didn't go on to define randomness, but he should have! If you think of a theory as a program that calculates the observations, the smaller the program is relative to the output, which is the observations, the better the theory is.

By the way, this is also what axioms do. I would say that axioms are the same idea. You have a lot of theorems or mathematical truth, and you're compressing them into a set of axioms. Now why is this good? Because then there's less risk. Because the axioms are hypotheses that you have to make, and every time you make a hypothesis you have to take it on faith and there's risk—you're not proving it from anything, you're taking it as a given, and the less you assume, the safer it is. So the fewer axioms you have, the better off you are. So the more compression of a lot of theorems, of a body of theory, into a small set of axioms, the better off you are, I would say, in mathematics as well as physics.

Okay, so this is this notion of lack of structure or randomness. You have to define it first! If I'm going to find randomness or lack of structure, lack of pattern, in pure mathematics, first I've got to say what do I mean by that. And I like to call this subject algorithmic information theory. It deals with this algorithmic information. Or you can call it complexity if you like, program-size complexity.

Algorithmic Information

The basic concept is to look at the size of the most concise program, the smallest program—I don't care about running time—it's the most concise program that calculates something. That's the number of bits I have to give a computer in order to get it to produce this object. That's my most concise algorithmic description of something, and that's how I measure its complexity, it's algorithmic information content, or its program-size complexity.

This is like recursive function theory: I don't care about run time—so this is very impractical! So in that sense also what I'm doing is 1930s stuff, with this one extra idea thrown in of program size, of looking at the size of programs.

So what happens when you start looking at the size of programs?—and then something is random if the smallest program that calculates it is the

Sometimes the reason you can't solve a mathematical problem isn't because you're not smart enough or you're not determined enough; it's because there is no solution because maybe the mathematical question has no structure, maybe the answer has no pattern, maybe there is no order or structure that you can try to understand in the world of pure mathematics. Maybe sometimes the reason that you don't see a pattern or structure is because there is no pattern or structure!

same size as it is, and there's no compression. So the whole idea is look at the size of computer programs; don't care about run time—if it takes a billion, billion years I don't care! Information is the only thing I'm thinking about, bits of information, size of computer programs. Okay?

So what happens when you start playing with this idea? What happens is, everywhere you turn, you get incompleteness and undecidability, and you get it in the worst possible way. For example, this happens with the first thing you want to do: You can never decide whether an individual string of digits satisfies this definition of randomness or not. Impossible! You can never calculate the program-size complexity of anything. You can never determine what the size of the smallest program is.

If you have a program that calculates something, that gives you an upper bound, its size is an upper bound on the program-size complexity of what it calculates. But you can never prove any lower bounds. And that's my first incompleteness result in this area, and I think Jack Schwartz got very excited about it.

In normal, practical, useful complexity theory where you talk about time rather than bits of information, lower bounds are much harder than upper bounds. To get lower bounds on complexity is much harder than getting upper bounds on complexity. Because if you find a clever algorithm you get an upper bound on the time it takes to calculate something; if you find a way to do it that's fast, you've shown that it can be done that fast. The problem is to show that you've gotten the fastest possible algorithm, that's much harder, right? But it can be done in some cases, within a class of possible algorithms. Well, in algorithmic information theory, you can't prove any lower bounds! And I had an article about this in 1975 in *Scientific American*.

The basic idea is that you can't prove any lower bounds on the program-size complexity of individual objects. So in particular even though most strings of digits satisfy this definition of randomness, they're incompressible in this sense, they're random in this sense of lack of structure—it turns out you can show easily that most objects satisfy this definition, they have no structure—if you look at all hundred digit numbers, almost all of them have no structure according to this definition, but you can never be sure in individual cases, you can never prove it in individual cases.

More precisely, there may be finitely many exceptions. With N bits of axioms you can determine all the objects of program-size complexity up to N. But that's as far as you can go.

And my worst incompleteness result, my very worst incompleteness result, where you have complete lack of structure in pure mathematics has to do with a number I defined called the halting probability.

Ω = halting probability

How is this number defined? It's very simple. Turing said you can't decide whether a program halts; there's no mechanical procedure for doing that. And I say, let's consider a real number Ω, which is the probability that a program generated by tossing a coin halts. So I'm averaging over Turing's halting problem, saying if I generate a program by coin tossing, what is the probability that it halts, with no time limit? So this will give me a real number that's determined if you tell me—there's a subscript—what's the programming language.

Ω_{computer} = halting probability of a computer

Once you decide, then Ω is a well-defined real number. Mathematically, it's not a very sophisticated thing! Compared to large cardinals, sophisticated mathematics, this is a fairly low-brow object.

However, it turns out this object Ω is maximally unknowable!

Ω is maximally unknowable

What is it that's maximally unknowable? Well, it's the digits or bits of this number. Once I fix the computer programming language, this halting probability is a specific real number that depends on the choice of computer, or the programming language in which I generate a program by coin tossing. So this becomes a specific real number, and let's say I write it out in binary, so I get a sequence of 0s and 1s—it's a very simple-minded definition. Well, it turns out these 0s and 1s have no mathematical structure. They cannot be compressed. To calculate the first N bits of this number in binary requires an N-bit program. To be able to prove what the first N bits of this number are requires N bits of axioms. This is irreducible mathematical information—that's the key idea.

Ω is irreducible information

This should be a shocking idea, irreducible mathematical information, because the whole normal idea of mathematics, the Hilbertian idea, the classical idea of mathematics, is that all of mathematical truth can be reduced to a small set of axioms that we can all agree on, that are "self-evident" hopefully. But if you want to determine what the bits of the halting probability Ω are, this is something that cannot be reduced to anything simpler than it is.

Ω has a mathematical definition with a rather simple structure once I specify the computer, or the programming language, I've even written out a program in LISP that calculates this number in a weak sense. You can't calculate this number. If you could calculate it, then it wouldn't be unknowable! You can get it in the limit from below, but it converges very, very slowly—you can never know how close you are—there is no computable regulator of convergence, there is no way to decide how far out to go to get the first N bits of Ω right. To get Ω in the limit from below, you just look at more and more programs, for more and more time, and every time you see that a K-bit program halts, that contributes $1/2^K$ to the halting probability.

$$\Omega = \sum_{p \text{ halts}} 2^{-|p|}$$

So the time you need to get the first N bits of Ω right grows like the longest possible finite run-time of an N-bit program, which is a version of the Busy-Beaver function.

So what's the precise definition of Ω? Generate a program by tossing a coin for each bit, that's independent tosses of a fair coin. The key point is that the program has to be "self-delimiting." The computer has got to ask for each bit one by one. Every time the computer says I want another bit of the program, you flip the coin. And the computer has to decide by itself that it has enough bits, that it has the whole program. The program has to be self-delimiting to define this probability measure correctly. So there's no blank to indicate where a program ends: A program has to indicate within itself how long it is with some trick, some coding trick. That's the technical issue to get this probability to be well-defined. That's the one technical point in my theory.

So this number Ω is a real number between 0 and 1. It's the probability that a program each of whose bits is generated by an independent toss of a fair coin eventually halts. And I'm fixing the programming language, I pick the universal Turing machine, there's a subscript, it's Ω_{UTM}, it's the halting probability of a particular universal Turing machine. And I actually pick a particular UTM that I programmed in LISP, just to fix the ideas. But you could do it with essentially any universal Turing machine with self-delimiting programs, it would work.

So Ω is maximally unknowable. This is a case where mathematical truth has no structure or pattern and it's something we're never going to know! So let me tell you what I've got here. What I've got here is maximum randomness—like independent tosses of a fair coin—in pure mathematics. In fact, I can even do it in elementary number theory, like Gödel did. I can make determining bits of Ω into an assertion about a diophantine equation.

The point is, here you've got a simple mathematical question—which is what is each bit of Ω, is the first bit 0 or 1, is the second bit 0 or 1, is the third bit 0 or 1—but the answers have no structure; they look like independent tosses of a fair coin, even though each answer is well-defined mathematically, because it's a specific bit of a specific real number and it has to be a 0 or a 1. In fact, we're never going to know: This is my version of independent tosses of a fair coin in pure mathematics. Even if you knew all the even bits of Ω, it wouldn't help you to get any of the odd bits. Even if you knew the first million bits, it wouldn't help you to get the next one. It really looks like independent tosses of a fair coin, it's maximally random; it has maximum entropy.

Physicists feel comfortable with randomness, but this is the black-or-white world of pure mathematics—how is this possible? How can it be? Each of these bits is well-defined; it's a specific 0 or a 1 because Ω is a specific real number once I fix the universal Turing machine or the programming language that I'm dealing with. But it turns out that the right way to think about each bit is that it's not black or white; it's not that it's a 0 or a 1. It's so well balanced, so delicately balanced, that it's grey!

Here's another way to put it. Let's go back to Leibniz. What's the idea of mathematics? The normal idea is that if something is true, it's true for a reason—Leibniz!—if something is true, it's true for a reason. Now in pure math, the reason that something is true is called a proof, and the job of the mathematician is to find proofs, to find the reason something is true. But the bits of this number Ω, whether they're 0 or 1, are mathematical truths that are true for no reason; they're true by accident! And that's why we will never know what these bits are.

In other words, it's not just that Hilbert was a little bit wrong. It's not just that the normal notion of pure mathematics is a little bit wrong, that there are a few small holes, that there are a few degenerate cases like "This statement is unprovable." It's not that way! It's much, much worse than that! There are extreme cases where mathematical truth has no structure at all, where it's maximally unknowable, where it's completely accidental, where you have mathematical truths that are like coin tosses, they're true by accident, they're true for no reason. That's why you can never prove whether individual bits of Ω are 0 or are 1, because there is no reason that individual bits are 0 or 1! That's why you can't find a proof. In other words, it's so delicately balanced whether each bit is 0 or 1 that we're never going to know.

So it turned out that not only Hilbert was wrong, as Gödel and Turing showed . . . . I want to summarize all of this. With Gödel it looks surprising that you have incompleteness, that no finite set of axioms can contain all of mathematical truth. With Turing incompleteness seems much more natural. But with my approach, when you look at program size, I would say that it looks inevitable. Wherever you turn, you smash up against a stone wall and incompleteness hits you in the face!

Program-size complexity and Ω and irreducible information make incompleteness seem inevitable.

So this is what I've been working on. Now what is the reaction of the world to this work?! Well, I think it's fair to say that the only people who like what I'm doing are physicists! This is not surprising because the idea came in a way from physics. I have a foreign idea called randomness that I'm bringing into logic, and logicians feel very uncomfortable with it. You know, the notion of program size, program-size complexity is like the idea of entropy in thermodynamics. So it turns out that physicists find this nice because they view it as ideas from their field invading logic. But logicians don't like this very much.

I think there may be political reasons, but I think there are also legitimate conceptual reasons, because these are ideas that are so foreign, the idea of randomness or of things that are true by accident is so foreign to a mathematician or a logician, that it's a nightmare! This is their worst nightmare come true! I think they would prefer not to think about it.

On the other hand, physicists think this is delightful! Because they remember well the crisis that they went through in the 1920s about randomness at the foundations of physics, and they say, it's not just us, we're not the only people who have randomness; pure math has it too, and they're not any better than we are!

To give an example of the attitude of physicists to my theory, it just so happens that this week I found out by chance . . . . There's an English magazine *New Scientist* that comes out every week; it's like an English version of *Scientific American,* except that it's a little livelier, it's a little more fun, and it comes out every week. And the current issue (the one that appeared February 26th, the next issue hasn't come out yet) of *New Scientist* has on its cover an article called "Random Reality." And if you open the issue and look at this article, it turns out to be an article about the work of two physicists—very speculative work. They're trying to get space and time, three- or four-dimensional spacetime, our world, to emerge from a random substratum underneath.

Go look at it if you like. There's a link on my Web site to this article, "Random Reality." Or get the *New Scientist.*

The reason that I mention this article is that these physicists say that their work was inspired by Gödel's and my work on the limits of logic; they're trying to absorb this stuff. They say that physicists were interested in Gödel's result, but they couldn't relate to it; it's not in terms that make sense to a physicist. But my work, they say, that makes sense to a physicist! It's not surprising: I got the idea by reading physics. So it makes sense to them because it's an idea that came from their field and is coming back to their field.

Actually, they don't use my definitions or my theorems at all, because I was asked to referee their paper, and I had to say that it really has nothing to do with me. My stuff is mentioned in the introduction because it helped to stimulate their work, but actually their work is in physics and has nothing to do with my area, which is algorithmic information theory.

But I think this is an interesting example of the fact that crazy ideas sometimes have unexpected consequences! As I said, formal systems did not succeed for reasoning, but they succeeded wonderfully for computation. So Hilbert is the most incredible success in the world, but as technology, not as epistemology.

And unexpectedly there are physicists who are interested in my notion of program-size complexity; they view it as another take on thermodynamical entropy. There's some work by real physicists on Maxwell's demon using my ideas; I mention this for those of you who have some physics background. But I must say that philosophers have not picked up the ball. I think logicians hate my work, they detest it! And I'm like pornography, I'm sort of an unmentionable subject in the world of logic, because my results are so disgusting!

So this is my story! To end, let me quote from a posthumous collection of essays by Isaiah Berlin, *The Power of Ideas,* which was just published: "Over a hundred years ago, the German poet Heine warned the French not to underestimate the power of ideas: philosophical concepts nurtured in the stillness of a professor's study could destroy a civilization." So beware of ideas, I think it's really true.

Hilbert's idea of going to the limit, of complete formalization, which was for epistemological reasons, this was a philosophical controversy about the foundations of mathematics—are there foundations? And in a way this project failed, as I've explained, because of the work of Gödel and Turing. But here we are with these complete formalizations, which are computer programming languages; they're everywhere! They pay my salary; they probably pay your salary . . . . Well, this is the School of Computer Science; it pays for all of this, right? Here we are!

So it worked! In another sense, it worked tremendously.

So I like to apologize in an aggressive way about my field. I like to say that my field has no applications, that the most interesting thing about the field of program-size complexity is that it has no applications, is that it proves that it cannot be applied! Because you can't calculate the size of the smallest program. But that's what's fascinating about it, because it reveals limits to what we can know. That's why program-size complexity has epistemological significance.

More seriously, I think the moral of the story is that deep ideas don't have a spin-off in dollars right away, but sometimes they have vastly unexpected consequences. I never expected to see two physicists refer to my stuff the way they did in "Random Reality." So who knows!

It's true that the computer pays for our salaries, but I think it's also true that there are a lot of fascinating impractical ideas out there, and sometimes when an idea is so beautiful . . . I've been having wonderful conversations with people here. Peter Lee told me over lunch, this idea is so beautiful, it's got to be right! Those are the ideas to watch out for! Those are the dangerous ones, the ones that can transform our society. This little idea of a Web, for example, of linking stuff into a Web! Or the idea of having completely artificial languages, because then it becomes mechanical to see what they mean . . . . Very dangerous ideas!

Thanks very much!

BIBLIOGRAPHY

1. Chaitin, G.J. Algorithmic Information Theory; Cambridge University Press, 1987.
2. Chaitin, G.J. Information, Randomness & Incompleteness; World Scientific, 1987.
3. Chaitin, G.J. Information, Randomness & Incompleteness, 2nd ed.; World Scientific, 1990.
4. Chaitin, G.J. Information-Theoretic Incompleteness; World Scientific, 1992.
5. Chaitin, G.J. The Limits of Mathematics; Springer-Verlag, 1998 [To be published in Japanese by Serendip].
6. Chaitin, G.J. The Unknowable; Springer-Verlag, 1999 [To be published in Japanese by Serendip].
7. I've recently finished programming my entire theory in LISP. This will eventually be a part of my book, *Algorithmic Information Theory in LISP* (in preparation).