CS 480 **Computational Theory**

The Ocean of Truth

From **Mind Tools** *By Rudy Rucker*

Many types of problems can be viewed as problems about deciding which numbers have some special property. Thus, if we imagine coding every sentence up as a number, then the problem of truth turns into the problem of deciding which numbers have the property of being code numbers of true sentences. We say that a property is "computable" if a computer can be programmed to decide whether or not any given number has the property in question. Using the notion of a general Turing machine with program P, we can make this quite precise. A property R of numbers is said to be computable if there is a program P such that

$$N \text{ has property } R \leftrightarrow P(N) = 1, \text{ and } N \text{ does not have property } R \leftrightarrow P(N) = 0.$$

In the last section we found that "being the code number of a Turing machine that halts" is not computable. Although the property of being a program that doesn't run forever is not computable, it is "listable." That is, there is a Turing machine that lists, in no particular order, all the programs that will eventually halt. Let me be more precise. A property R of numbers is said to be listable if there is a program P such that

$$N \text{ has property } R \leftrightarrow P(K) = N \text{ for some number } K.$$

I say that the set of all programs that halt is listable because I can imagine setting up a machine that, given any K, tests one program after another, for longer and longer periods of time, until it finds the first K programs that halt and then prints out the code number of the K^{th} of these programs.

Interestingly enough, the property of being the code number of a true sentence about numbers is neither listable nor computable. Not only can no program be designed to differentiate between true and false sentences about numbers, no program can be designed to list all the true sentences with no false sentences included.

If a property is computable, this means that there is a mechanical program P that can look at any N and give a yes or no answer as to whether or not N has the property. "Being a prime number" is an example of a computable property, for given a number N, I can try dividing N by each of the numbers less than N. If N is prime, then none of these numbers will go into N, and if N is not prime then some number (other than one) will go into N. "Being the code number of a tautology" is another example of a computable property. Given the code number of a compound sentence S, I can set up a truth table and check if S has to be true regardless of the truth or falsity of its various clauses.

So a number property R is computable if I can build a Turing machine that reliably detects the presence or absence of property R. Metaphorically speaking, the property of being a person carrying metal is a computable one—airport metal detectors can decide, for any given person, whether or not that person is carrying metal. If metal is present, the detector gate lights a red light; if there is no metal, the detector lights a green light.

If a property is listable, this means that there is a mechanical program P that can list all the numbers having the property. "Being a code number of a theorem of the theory M" is an example of a listable property. Given a theory M, I can start systematically deriving all of M's consequences. If the number N happens to code a theorem of M, then I will eventually get to that theorem, and I will know that N has the property of naming a theorem. Some short theorems have very long proofs, so even if I have not yet proved theorem N, I always have to wonder if I might prove it later.

Let us imagine a situation in which being "saved" is a listable property. Suppose that, at the end of time, an infinite sea of humanity is standing in front of the pearly gates. Saint Peter is a very kind guy, and he never looks at someone and says, "You're damned, go to Hell." Instead, he spiels out an infinite list of possible qualifications for salvation. "OK, if you ever saved someone's life, welcome to heaven. People who gave ten dollars to a bum on the street, come on in. Anyone who was born in 1946 come in, too; I like that year. All right. If you died poor, you can get in. Any skin divers? All skin divers are saved." On and on he goes, through an endless list of possible qualifications for salvation. If you're saved, you eventually find out, but if you're damned, you stand there, uncertain forever.

The distinction between computable properties and listable properties may still seem a bit obscure. Why do I bring such a distinction up at all? The reason is that for most of the formal systems of logic we are interested in, the set of all the systems' theories is listable, but not computable. That is, for any decent logical system M, we can list all of M's theorems, but there is no finite mechanical way to look at a sentence S and automatically know whether or not S is a theorem. That there is no such finite program was proved, as I have mentioned, by Alonzo Church in 1936, and is known as Church's theorem.

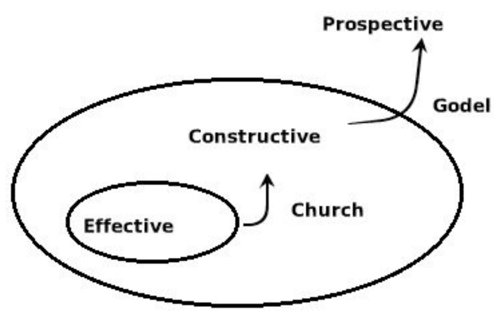

Figure 1: Church's Theorem proves that some constructive properties are not effective. Gödel's theorem tells us that some prospective properties are not constructive.

If we think of all known mathematics as codified in a theory M, the property of being an *axiom* of M will be a computable property. The axioms we use can be grouped into a finite number of simple schemata; and checking a candidate axiom S against the schemata is a mechanical procedure that yields a definite Yes or No in a finite amount of time.

Deciding if a statement is provable from M is more difficult, as we have no way of predicting how long a given statement's proof might be. Church's result tells us that, indeed, the property of being a *theorem* of M is listable but not computable.

Even more elusive is the property of being a *true sentence* in the mathematical language of M. Gödel's theorem tells us that this property is not even listable.

Thus we see there are three categories of increasingly complex kinds of property: computable properties, listable properties, and properties that are neither computable nor listable. Suppose that we call a property of the third type "prospective." The set of axioms is computable, the set of theorems is listable, and the set of truths is prospective.

The term prospective is taken from a 1952 paper by John Myhill called "Some Philosophical Implications of Mathematical Logic: Three Classes of Ideas." In this paper Myhill did something very unusual for the uptight formalistic 1950's—he demonstrated that Church's and Gödel's theorems can be thought of in a metaphorical way.

Church and Gödel tell us that axioms, theorems, and truth lie at three different levels of complexity. Myhill points out that the same three-level split can be found in many other places.

Consider a written page S, such as this page, a page covered with letters, spaces, and punctuation marks. Deciding if this page is a passage of grammatical English is a computable matter. One has only to check if the words are all dictionary words, and if the words' parts of speech are such that they can be fit into conversational sentence formats.

Even though the page S may be grammatical English, it may very well be meaningless to you. This could be either because the page is gibberish, or because you are unfamiliar with the ideas the page refers to. You are the only real judge of whether S is meaningful to you. As a rule, the longer you live, the more different pages S will seem meaningful. But it is impossible to predict in advance which pages will eventually become meaningful to you. The property of being meaningful is listable but not computable.

By the same token, the question of whether a page S might ever be something you'd want to *write* is a listable one. An author's writings arise, perhaps, in a more or less mechanical way from the endless churning of his or her brain, but there is no way to predict years in advance whether or not the brain in question will eventually output the page S. How many times, as a writer, have I thought, "I can't believe I'm saying this!" The same holds true for utterances. That is, the property of a string's being a sentence you might eventually want to say is listable (for all your life you are orally "listing" these sentences), but surely not computable.

Higher properties—such as truth, beauty or virtue—are prospective. There is no fixed rule or token by which we can recognize the true or the beautiful or the good: these human ideals are not computable. Nor is there any kind of program or attitude that will enable any individual person or school to produce all truth or all beauty or all goodness: our highest goals are not to be exhausted by the logical working out of any single system.

Church's theorem is a metaphor for the first fact, the fact that no simple test can give Yes or No answers for important questions. And Gödel's theorem is a metaphor for the second fact, the fact that no logical program can hope, even in the limit, to answer all the questions.

After the work of Turing, Church, and Gödel, the old dream of capturing all truth in a finite logical net can be seen to be thoroughly bankrupt. Turing's analysis of computation suggests that every finitely given logical system (including human beings) is subject to the theorems of Gödel and Church. Gödel's theorem tells us that no programmatic method can generate all truth; while Church's theorem tells us that we are unable even to predict the consequences of the programs we do devise. Is this a cause for despair? Not really. It is rather, I would say, a cause for joy.

A world with no Gödel's theorem would be a world where every property is listable—for any kind of human activity, there would be a programmatic description of how to carry it out. In such a world it would be possible to learn a hard and fast formula for "how to be a scientist." It would just be a matter of learning the tricks of the trade.

A world with no Gödel's theorem and no Church's theorem would be a world where every property is computable—for any kind of human activity, there would be a fixed code for deciding if the results were good. In such a world an Academy could pass judgment on what was art and what was science. Creativity would be a matter of measuring up to the Academy's rules, and the Salon des Réfuses would contain only garbage.

But if there is one thing art history teaches us it is this: all tricks of the trade wear thin, and it's a good idea to keep an eye on the Salon des Réfuses.

Our world is endlessly more complicated than any finite program or any finite set of rules. You're free, and you're really alive, and there's no telling what you'll think of next, nor is there any reason you shouldn't kick over the traces and start a new life at any time.

From *Mind Tools*, pages 243-247