CDM

(Semi) Decidability

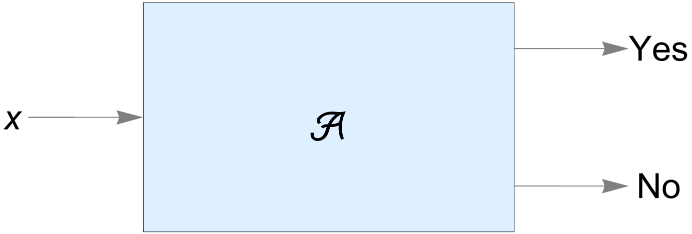
Klaus Sutner

Carnegie Mellon University

Fall 2024

Decidability

Informally, a problem is decidable if there is a decision algorithm $\mathcal{A}$ that returns Yes or No depending on whether the input has the property in question.



Formally . . .

We can easily model this in terms of computable functions:

Definition

A set $R \subseteq \mathbb{N}^k$ is decidable if the characteristic function char_R is computable.

A decision problem is decidable if the set of Yes-instances is decidable.

In keeping with the old recursive/partial recursive nomenclature, a decideable set is also called recursive.

Generalization

In other words, a set R is decideable iff its characteristic function is computable (i.e., it belongs to the clone of computable functions).

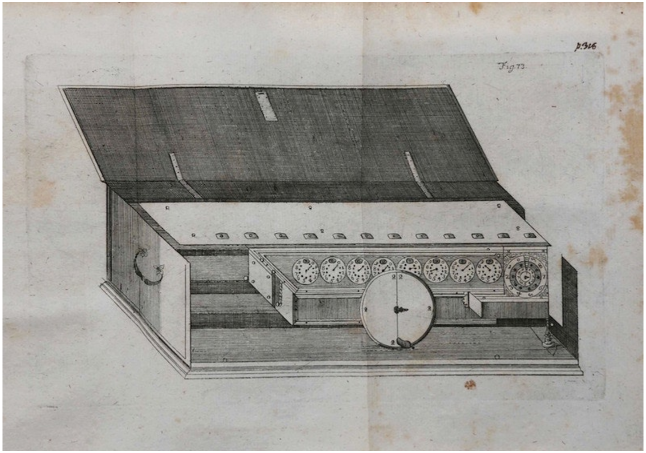
We could replace computable functions by any other clone $\mathcal{C}$ and get a notion of $\mathcal{C}$-decidable.

For example, we could talk about polynomial time decidable problems or logarithmic space decidable problems. Note that this requires closure under composition, it does not work for an arbitrary collection of functions.

Gottfried Leibniz (1646–1716)

In a way, the idea of decidability can be traced back to Leibniz.

A Calculator



Moon Shot

- ars inveniendi: discover all true scientific statements.

> I don't pay much attention to specific discoveries. What I most desire is to perfect the Art of Invention, and to provide methods rather than solutions to problems, for one single method comprises an infinity of solutions.

Moon Shot II

- ars iudicandi: determine whether a given scientific statement is true

> It is obvious that if we could find characters or signs suited for expressing all our thoughts as clearly and as exactly as arithmetic expresses numbers or geometry expresses lines, we could do in all matters insofar as they are subject to reasoning all that we can do in arithmetic and geometry. For all investigations which depend on reasoning would be carried out by transposing these characters and by a species of calculus.

Decision Algorithms

Decision problems have been around since the day of the flood: one is interested in checking whether a number is prime, whether a polynomial is irreducible, whether a polygon is convex, and so on. Gauss certainly understood the computational difficulty of primality checking quite well.

But Leibniz seems to have articulated the need for general decision algorithm more clearly than anyone before him. Alas, he was ahead of his time, nothing came of his ideas for a while.

The first big splash came in 1900, when Hilbert presented his famous list of 23 open problems at the International Congress of Mathematicians in Paris.

Hilbert's List

Hilbert's list was enormously influential throughout the 20th century.

Some of Hilbert's Problems

1. Prove the Continuum Hypothesis. Well-order the reals.
2. Prove that the axioms of arithmetic are consistent.

   . . .

8. Prove the Riemann Hypothesis.

   . . .

10. Given a Diophantine equation with any number of unknown quantities and with rational integral numerical coefficients: to devise a process according to which it can be determined by a finite number of operations whether the equation is solvable in rational integers.

The Mother of all Decision Problems

The Entscheidungsproblem is solved when one knows a procedure by which one can decide in a finite number of operations whether a given logical expression is generally valid or is satisfiable. The solution of the Entscheidungsproblem is of fundamental importance for the theory of all fields, the theorems of which are at all capable of logical development from finitely many axioms.

D. Hilbert, W. Ackermann
Grundzüge der theoretischen Logik, 1928

A Hedge

Note that Hilbert and Ackermann hedge a tiny little bit: the decision procedure for the Entscheidungsproblem needs to work only for areas where the fundamental assumptions can be finitely axiomatized.

The logician J. Herbrand[†] pointed out that

> In a sense, the Entscheidungsproblem is the most general problem of mathematics.

[†]Based on a suggestion by Herbrand, Gödel developed a model of computation that uses recursion over multiple variables.

Aside: Hardness

Both Leibniz and Hilbert were asking for too much; in particular, there is no algorithm to solve the Entscheidungsproblem.

But we can scale back a bit: try to solve the Entscheidungsproblem only for a small domain. For example, only worry about propositional logic (rather than full first-order). In this capacity, the Entscheidungsproblem is an excellent source of difficult problems in complexity theory.

For suitable versions of the Entscheidungsproblem, instead of undecidability, we get computational hardness. For example, for $\mathbb{NP}$, we only need Boolean formulae of type

$$\exists\, x_1, \ldots, x_n\ \varphi(x_1, \ldots, x_n)$$

Closure Properties

Lemma

The decidable sets are closed under intersection, union and complement. In other words, the decidable sets form a Boolean algebra.

The proof is exactly the same as for primitive recursive relations.

And, just like for primitive recursive relations we get a slightly better result: the Boolean algebra is effective in the sense that we can compute the algebraic operations (assuming the sets are represented by an index of their characteristic functions).

We have a computable structure, not just an abstract algebra.

Counting

Theorem

There are undecidable decision problems.

Proof.

There are uncountably many subsets of $\mathbb{N}$ (to be precise: $2^{\aleph_0}$). But there are only countably many register machines (recall our coding machinery).

Hence, almost all problems $A \subseteq \mathbb{N}$ are not decidable. $\square$

Note that this argument, like others imported from set theory, leaves a bitter after-taste: it does not produce any concrete undecidable problem.

Going Concrete

First off, “almost all” is meant purely set-theoretically. It might well be that all the interesting problems are decidable an no one cares about the uncountably many other guys.

At the very least, we would like a concrete undecidable problem, and we really would like examples of undecidable problems that are motivated by concrete problems in math and CS.

We already know that, regardless of what general model of computation one uses, there is always the undecidable question of Halting—which we can use to construct more interesting examples.

Halting

There are several equally reasonable versions of Halting:

Problem:	**Full Halting**
Instance:	Index $e \in \mathbb{N}$, an argument $x \in \mathbb{N}$.
Question:	Does register machine M_e halt on input x?

For technical reasons, the following slightly artificial version of Halting is also very popular:

Problem:	**Halting**
Instance:	Index $e \in \mathbb{N}$.
Question:	Does register machine M_e halt on input e?

Another Halting

We can also get rid of the input altogether: the machine can start with a precomputation that produces the input for the main computation.

Problem:	**Pure Halting**
Instance:	Index $e \in \mathbb{N}$.
Question:	Does register machine M_e halt?

In this case, all registers are zeroed out before the computation starts. Again, this is essentially the same as the other two versions. Arguably, this is the most elegant variant.

Halting is Undecidable

A standard diagonalization argument shows that Halting cannot be solved by a register machine.

Theorem

The Halting Problem is undecidable.

And, of course, we could do this in any model of computation, there always is a universal "machine."

Proof, Informal

Assume Halting is decidable. Then the following program produces a contradiction:

```
// impossible function

    if( halt(z,z) )
        return  eval(z,z) + 42;
    else
        return 0;
```

Here `halt(z,z)` is the Halting tester that exists by assumption, and `eval(z,z)` is the universal RM: run M_z on input z.

So this is the standard construction that turns a partial arithmetic function into a total one (the old f and $f_\perp$ trick).

Proof, Formal

Suppose Halting (the one-argument version) is decidable.

Then we can define a function g by cases:

$$g(z) \simeq \begin{cases} \{z\}(z) + 1 & \text{if } \{z\}(z) \downarrow, \\ 0 & \text{otherwise.} \end{cases}$$

From the description, this function g is computable. Hence, g has some index e, which promptly produces a contradiction: $g(e) = g(e) + 1$.

Essentially we are dealing again with our old eval problem, the issue that forces us to allow partial functions in the first place.

Diagonalization

Somewhat surprisingly, both arguments are based on diagonalization, originally invented by Cantor in the context of set-theory to establish the uncountability of the reals.

The very sametool also turns out to be of central importance in the computational universe, in recursion theory as well as complexity theory.

The Next Step

Decidability makes direct algorithmic sense, we are trying to test for certain properties in a computational manner.

The Halting Problem fails to be decidable, so it is natural to look for a larger class of problems that accommodates Halting. We want a positive characterization, not just a negative one.

A closer look shows that there is indeed a weakerm natural property, called semidecidability, that encompasses Halting, and that is arguably more fundamental and ultimately more important than decidability.

Semidecidability

We need to generalize decidability just a little bit:

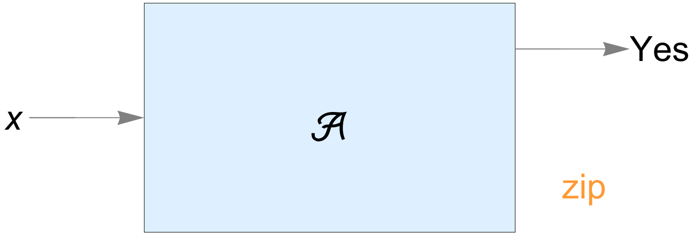
Definition

A set $A \subseteq \mathbb{N}^k$ is semidecidable if there is a register machine that, on input x, halts if $x \in A$, and fails to halt otherwise.

We will call this a semidecision procedure. As one might suspect, the term semi-recursive is also used.

You can think of this as a broken decision algorithm: if the answer is Yes, the algorithm works properly and stops after finitely many steps. But, if the answer is No, it just keeps running forever, it never produces a result.

In Pictures



Just to be clear: zip is not output, it just means "there is no output."

Quoi?

Note that Halting is the critical natural example of a semidecidable problem: we can determine convergence in finitely many steps, but divergence takes forever. This is exactly the idea behind an unbounded search: we find a witness in finitely many steps if there is one, but otherwise we keep searching forever.

Here is the critical connection between decidability and semidecidability:

Lemma

A set is decidable iff the set and its complement are both semidecidable.

Proof

Clearly, A decidable implies that both A and $\overline{A}$ are semidecidable.

But the opposite direction is far from trivial: we have two semidecision procedures $\mathcal{A}_0$ and $\mathcal{A}_1$, but we don't know which one is going to halt. So we cannot simply run, say, $\mathcal{A}_0$ first.

The way around this problem is to run both procedures in parallel on the given input: we combine two computations into one, alternating steps. We stop as soon as one of the sub-computations terminates.

Intuitive, this no surprise; every operating system does this. But it actually is another fundamental property of computation: we can interleave two computations into a single one.

Closure Properties

Lemma

The semidecidable sets are closed under intersection and union.

Proof.

For intersection we can simply run the two semidecision-procedures sequentially.

But for union we again need to interleave two computations.

□

Note: We do not have closure under complement in general: Halting and the last lemma prohibit that.

Kleene Normal Form

Here is a closer look at Halting and semidecidability. Suppose we have a universal register machine $\mathcal{U}$ producing an enumeration $\{e\}$ of all computable functions.

We claim that there is a primitive recursive relation $T(e, x, t)$ and a primitive recursive function D (in fact, both T and D are quite straightforward) such that

$$\{e\}(x)\downarrow \iff \exists t\ T(e,x,t)$$

$$\{e\}(x) \simeq D(\min\big(t \mid T(e,x,t)\big)$$

One can think of t as the number of steps required to complete the computation of machine e on input x.

Kleene's T Predicate

$T(e, x, t)$

- e the index of a register machine M
- x an input argument for M
- t a witness for a halting computation of M on input x.

The witness is usually the (sequence number that codes) a sequence of configurations $C_0, C_1, \ldots, C_n$ of M.

Since we have an alleged witness, T is easily decidable and primitive recursive.

Moreover, given the right t, it is easy to read off the output of the computation (that's D's job).

Still Unhappy?

Halting may well seem like a somewhat unsatisfactory example of an undecidable problem: it's a perfect case of navel gazing. Certainly it does not deal with a question at least superficially unrelated to computability.

Actually, anyone who has ever written a complicated program in a language like C would have to admit that Halting is really quite natural.

But how about undecidable problems that are of independent interest? Perhaps something that was studied even before the concept of an algorithm was invented?

Warning: Treacherous Arithmetic

Basic arithmetic on the natural numbers may seem fairly straightforward; tedious on occasion, but not truly complicated.

Wrong, wrong, wrong. Consider the numbers

$$n^{17} + 9 \qquad \text{and} \qquad (n{+}1)^{17} + 9$$

where $n \geq 0$. They turn out to be coprime up until we hit the first counterexample

$$n = 8424432925592889329288197322308900672459420460792433$$

This is rather large, about 8.42×10^{51}.

Where the hell does this huge number come from?

Hilbert's 10th Problem

Perhaps the most famous example of an undecidability result in mathematics is Hilbert's 10th problem, the insolubility of Diophantine equations: we have a polynomial equation with integer coefficients:

$$P(x_1, x_2, \dots, x_n) = 0$$

The problem is to determine whether such an equation has an integral solution.

Theorem (Matiyasevic, Davis, Putnam, Robinson)

It is undecidable whether a Diophantine equation has a solution in the integers.

Examples

Pythagorean triples: it is not hard to classify all the solutions of

$$x^2 + y^2 - z^2 = 0$$

But tackling Fermat triples

$$x^k + y^k - z^k = 0$$

for $k > 2$ was one of the central problems of number theory and it took more than 350 years to show that solutions only exist for $k = 1, 2$.

The Proof

The proof of undecidability of Diophantine equations is way too complicated to be presented here, but the main idea is a reduction using very clever coding tricks:

> *Show that decidability of Hilbert's 10th problem implies decidability of the Halting problem.*

More precisely, call a set $A \subseteq \mathbb{Z}$ Diophantine if there is a polynomial P with coefficients over $\mathbb{Z}$ such that

$$a \in A \iff \exists\, x_1, \ldots, x_n \in \mathbb{Z} \left(P(a, x_1, \ldots, x_n) = 0 \right).$$

This condition is rather unwieldy, it is usually fairly difficult to show that a particular set is in fact Diophantine.

Diophantine Sets

We want to describe $a \in \mathbb{Z}$ with some particular property:

Even: $a = 2x$.

Non-zero: $a\,x = (2y+1)(3z-1)$.

Non-negative: $a = x_1^2 + x_2^2 + x_3^2 + x_4^2$ (Lagrange's theorem)

Similarly we can show closure under intersection (sum of squares) and union (product). But note that complements don't work in general.

Again: Semidecidability

It turns out that exactly the semidecidable sets are Diophantine. In particular, the Halting Set is Diophantine, and so it must be undecidable whether an integer polynomial has an integral solution.

It is clear that every Diophantine set is semidecidable: given a, we can simply enumerate all possible $\boldsymbol{x} \in \mathbb{Z}^n$ in a systematic way, and compute $P(a, \boldsymbol{x})$.

If we ever hit 0, we stop; otherwise we run forever.

Surprisingly the opposite direction also holds, but this is much, much harder to show.

Details

First M. Davis was able to show that every semidecidable set A has a Davis normal form: there is a polynomial such that

$$a \in A \iff \exists\, z\, \forall\, y < z\, \exists\, \boldsymbol{x}\, \big(P(a, \boldsymbol{x}, y, z) = 0\big)$$

Davis, Putnam and Robinson then managed to remove the offending bounded universal quantifier at the cost of changing P to an exponential polynomial (containing terms x^y).

Lastly, Matiyasevic showed in 1970 how to convert the exponential polynomial into an ordinary one.

Bounding Roots

Note that if we could produce a computable bound on the size of a possible root we could use brute-force search to determine whether one exists (in principle, in reality we die an exponential death).

Consider a univariate polynomial $p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots a_0$ of degree n (so $a_n \neq 0$).

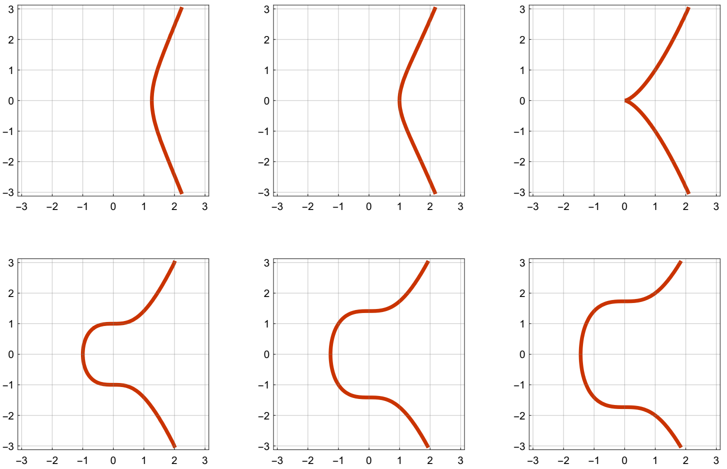
By rearranging terms a bit and using the standard properties of inequalities over the reals we find that for any root x

$$|x| \leq n \cdot \frac{a_{\max}}{|a_n|}$$

where $a_{\max} = \max(|a_{n-1}|, \dots, |a_0|)$. Done by search.

Elliptic Curves

How about a simple elliptic curve, say, $y^2 = x^3 + c$? Here is $-2 \leq c \leq 3$.



Discrete Case

For real solutions, this scenario is trivial. But we are looking for integral solutions, and then things get incredibly messy.

It is known that the least integral solution, if it exists, is bounded by

$$\exp((10^{10}|c|)^{10000})$$

So, in principle, this is decidable by brute force search.

In practice, nothing moves; even if we were to allow nondeterminism. Elliptic curves turn out to be useful in cryptography.

Cubes

How about sums of three cubes? How hard could that be?

$$x^3 + y^3 + z^3 = c$$

The smallest solutions for $c = 74$ and $c = 33$ are

$$74 = -284650292555885^3 - 66229832190556^3 + 283450105697727^3$$
$$33 = 8866128975287528^3 - 8778405442862239^3 - 2736111468807040^3$$

A Horror: It is not known that this problem is decidable. Worse, it may well be that c is the sum of 3 cubes iff $c \neq \pm 4 \pmod 9$. But that's an open problem.

This just in . . .

Andrew Sutherland at MIT and Andrew Booker at U Bristol used over a million hours of compute time on the Charity Engine to find

$$42 = -80538738812075974^3 + 80435758145817515^3 + 12602123297335631^3$$

Bounding Roots, II

So it would not be unreasonable to think that integer roots of $P(x_1, \ldots, x_n)$ can be bounded by some rapidly growing but computable function of n, the degree d of P, and the largest coefficient c.

Perhaps something insanely huge, using the Ackermann function:

$$A\left(n!\,|c|^d, n^{n^{d+42}}\right)$$

would work?

If not, try an even higher level of the Ackermann hierarchy or one of Friedman's monsters?

Different Rings

Note that the choice of $\mathbb{Z}$ as ground ring is important here. We can ask the same question for polynomial equations over other rings R (always assuming that the coefficients have simple descriptions).

- $\mathbb{Z}$: undecidable
- $\mathbb{Q}$: major open problem
- $\mathbb{R}$: decidable
- $\mathbb{C}$: decidable

Decidability of Diophantine equations over the reals is a famous result by A. Tarski from 1951, later improved by P. Cohen.

One Implication

It is true that an algorithm for Diophantine equations over $\mathbb{Z}$ would produce an algorithm for Diophantine equations over $\mathbb{Q}$.

Consider $P(\boldsymbol{x}) = 0$ with $\boldsymbol{x} \in \mathbb{Q}$. This is equivalent to

$$\exists\, \boldsymbol{y} \in \mathbb{Z}^n, \boldsymbol{z} \in \mathbb{N}_+^n \left(P(\boldsymbol{y}/\boldsymbol{z}) = 0\right)$$

By multiplying with powers of the z_i, the RHS turns into a Diophantine equation over $\mathbb{Z}$, albeit with more variables.

Alas, this is exactly the wrong direction, we want to show that an algorithm over $\mathbb{Q}$ produces an algorithm over $\mathbb{Z}$.

Hard Polynomials

Since we can encode arbitrary semidecidable sets as Diophantine equations, we can in particular encode universal sets.

That means that there is a single polynomial with parameter a for which the question

$$\exists\, x_1, \ldots, x_n\, P(a, x_1, \ldots, x_n) = 0$$

is already undecidable.

As one might suspect, there is a trade-off between the degree d of such a polynomial and its number of variables n. Here are some known (d, n) pairs that admit universal polynomials:

$$\begin{aligned} &(4, 58), (8, 38), (12, 32), \ldots, \\ &(4.6 \times 10^{44}, 11), (8.6 \times 10^{44}, 10), (1.6 \times 10^{45}, 9) \end{aligned}$$

Jones, Sato, Wada, Wiens Polynomial

$$
\begin{gathered}
(k+2)\Big(1-[wz+h+j-q]^2-[(gk+2g+k+1)(h+j)+h-z]^2- \\
[16(k+1)^3(k+2)(n+1)^2+1-f^2]^2-[2n+p+q+z-e]^2- \\
[e^3(e+2)(a+1)^2+1-o^2]^2-[(a^2-1)y^2+1-x^2]^2- \\
[16r^2y^4(a^2-1)+1-u^2]^2-[n+l+v-y]^2-[(a^2-1)l^2+1-m^2]^2- \\
[ai+k+1-l-i]^2-[((a+u^2(u^2-a))^2-1)(n+4dy)^2+1-(x+cu)^2]^2- \\
[p+l(a-n-1)+b(2an+2a-n^2-2n-2)-m]^2-[q+y(a-p-1)+ \\
s(2ap+2a-p^2-2p-2)-x]^2-[z+pl(a-p)+t(2ap-p^2-1)-pm]^2\Big)
\end{gathered}
$$

This polynomial P has 26 variables and degree 25. $P(\mathbb{N}^{26})\cap\mathbb{N}$ is the set of prime numbers. Note the factor $(k+2)$ up front.

A Real Math Problem

Arguably the most important, longstanding open problem in math is the Riemann Hypothesis.

In its original form, RH looks like a solid chunk of complex analysis. You start with the Riemann zeta function

$$\zeta(s) = \sum_{n \geq 1} \frac{1}{n^s}$$

where $s \in \mathbb{C}$, $\mathrm{Re}(s) > 1$.

The condition $\mathrm{Re}(s) > 1$ is needed to ensure convergence of the series (think about $s = 1$).

Better Riemann

Using a technique known as analytic continuation, one can then extend ζ to a function of type $\mathbb{C} \to \mathbb{C}$ with a pole at $s = 1$.

The reason ζ is so hugely important is that it is closely connected to prime numbers, a fact already known to Euler.

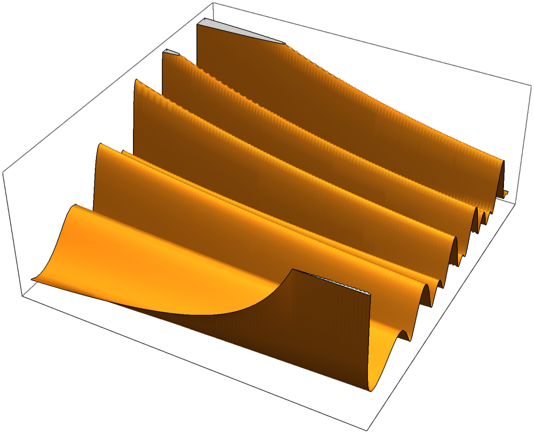
$$\zeta(s) = \prod_p \frac{1}{1 - p^{-s}}$$

where the product is over all primes.

A better understanding of ζ would have huge implications for the distribution of primes. For example, a more general form of RH, implies that a particular primality test runs in polynomial time (G. Miller).

Visualization

Since ζ is a complex function, it is rather hard to draw.



The modulus of $\zeta(s)$ for $0 \leq \mathrm{Re}(s) \leq 1$ and $0 \leq \mathrm{Im}(s) \leq 50$.

Roots

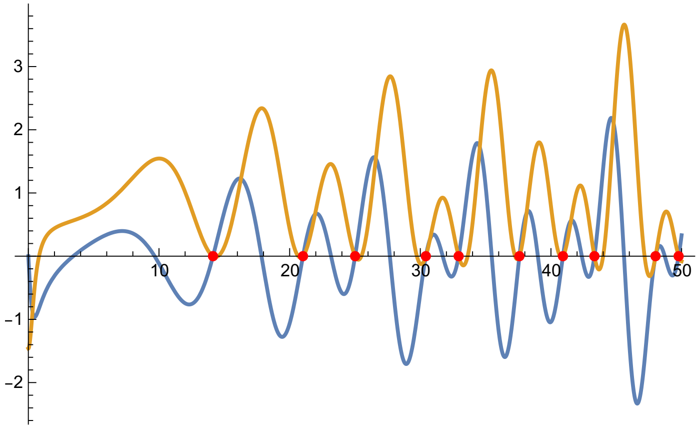
One can show that $\zeta(-2k) = 0$ for all positive integers k (trivial zeros).

Alas, there are other, non-real roots that are much harder to understand. Here is the big conjecture, proposed by Bernhard Riemann in 1859:

All non-real roots s have the property $\mathrm{Re}(s) = 1/2$.

Note that this statement seems to require considerable horse-power: first we need to define the analytic extension of a power series, then we want to argue about some of roots of the resulting function.

Complex Roots



Complex and real parts of $\zeta(1/2 + \mathbf{i}\,t)$ for $0 \leq t \leq 50$.

The Computability Sledgehammer

So far, we are dealing with complex analysis. The next step is to reduce all this complicated material to computation.

A formula of arithmetic is said to be Π_1 if it can be written down by using just one universal quantifier over $\mathbb{N}$, plus bounded quantifiers, propositional logic and standard arithmetic.

$$\Theta \equiv \forall\, n\, \phi(n)$$

where ϕ is a formula of basic arithmetic. The relation $\phi(n)$ is in particular primitive recursive and easy to check.

But checking Θ itself is more problematic: on the face of it, we have to perform an infinite computation, we have try out all possible values of n.

It's Π_1

Big Surprise: the Riemann Hypothesis is Π_1.

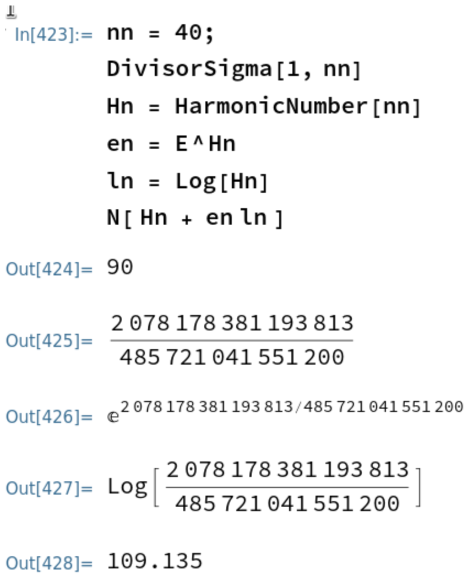
This may sound patently wrong, but can be shown with enough effort: let $H_n = \sum_{k \leq n} 1/k$ be the nth harmonic number, and $\sigma(n)$ the divisor function (total sum of all divisors of n). These are all primitive recursive.

The RH is equivalent to: for all n,

$$\sigma(n) \leq H_n + e^{H_n} \log H_n$$

As written, this looks like real arithmetic involving e and $\log$. In the actual argument, everything has to be rephrased in terms of rational numbers, and one needs to be very careful with error estimates.

Example: $n = 40$



```
In[423]:= nn = 40;
          DivisorSigma[1, nn]
          Hn = HarmonicNumber[nn]
          en = E^Hn
          ln = Log[Hn]
          N[ Hn + en ln ]

Out[424]= 90
```

$$\text{Out[425]=}\ \frac{2\,078\,178\,381\,193\,813}{485\,721\,041\,551\,200}$$

$$\text{Out[426]=}\ e^{2\,078\,178\,381\,193\,813/485\,721\,041\,551\,200}$$

$$\text{Out[427]=}\ \mathrm{Log}\left[\frac{2\,078\,178\,381\,193\,813}{485\,721\,041\,551\,200}\right]$$

```
Out[428]= 109.135
```

The Riemann Register Machine

Based on this inequality, one can now construct a register machine $\mathcal{M}$ that runs through a (potentially infinite) loop and tries to check the inequality for all n.

If $\mathcal{M}$ finds a counterexample, it stops and returns No. Otherwise $\mathcal{M}$ runs forever.

Claim: $\mathcal{M}$ halts iff the Riemann Hypothesis is false.

There even are estimates on how large $\mathcal{M}$ would need to be (though existing work uses Turing machines rather than register machines; less than $10,000$ states).

Pushing Things

Similarly we can design a register machine $\mathcal{Z}$ that systematically enumerates all first-order logic theorems provable in Zermelo-Fraenkel with Choice. This works since the axioms of ZFC are easily decidable.

Now suppose $\mathcal{Z}$ is set up to halt when it finds a proof of $\emptyset = \{\emptyset\}$, otherwise it runs forever.

By Gödel's incompleteness theorem, and assuming that ZFC is consistent, we cannot prove in ZFC that $\mathcal{Z}$ never halts, nor can we prove that it halts.

So ZFC is too weak to say anything about the Halting behavior of $\mathcal{Z}$.

Halting is seriously hard.