Incompleteness, the universal algorithm, and arithmetic potentialism

Kameryn J. Williams

University of Hawaiʻi at Mānoa

Talk Math With Your Friends
2020 September 17

A very accurate and nuanced early history of the foundations of computation

Find an algorithm to solve the *Entscheidungsproblem**.

No.

* (Given a logical formula determine whether it is true in all structures.)

In a bit more detail

- The strategy to show an algorithm solves the *Entscheidungsproblem* is straightforward: exhibit the algorithm and check it does what you want.
- But how to show that there can be no such algorithm?

In a bit more detail

- The strategy to show an algorithm solves the *Entscheidungsproblem* is straightforward: exhibit the algorithm and check it does what you want.
- But how to show that there can be no such algorithm?
- Need an abstract notion of algorithm so that you can do math with this definition.
- Alonzo Church (1936), Alan Turing (1936), and others gave formalizations, which turn out to be equivalent.
- And since then there has been an explosion in equivalent characterizations, e.g. (an idealized version of) your favorite programming language.

In a bit more detail

- The strategy to show an algorithm solves the *Entscheidungsproblem* is straightforward: exhibit the algorithm and check it does what you want.
- But how to show that there can be no such algorithm?
- Need an abstract notion of algorithm so that you can do math with this definition.
- Alonzo Church (1936), Alan Turing (1936), and others gave formalizations, which turn out to be equivalent.
- And since then there has been an explosion in equivalent characterizations, e.g. (an idealized version of) your favorite programming language.
- An advantage to giving a talk in 2020 is that computers are so ubiquitous I don't need to give you the formal definition of a Turing machine (TM).

Turing reduced the *Entscheidungsproblem* to the halting problem

Theorem (Turing)

There is no Turing machine which accepts as input a TM p and input n for p and determines whether or not p with halt on n and produce an answer.

Turing reduced the *Entscheidungsproblem* to the halting problem

Theorem (Turing)

There is no Turing machine which accepts as input a TM p and input n for p and determines whether or not p with halt on n and produce an answer.

- Hard part! Turing showed that TMs are powerful enough to do computations involving other TMs. Indeed, he showed there is a universal machine which can simulate any TM.

- Easy part! Do a diagonalization argument.

Turing reduced the *Entscheidungsproblem* to the halting problem

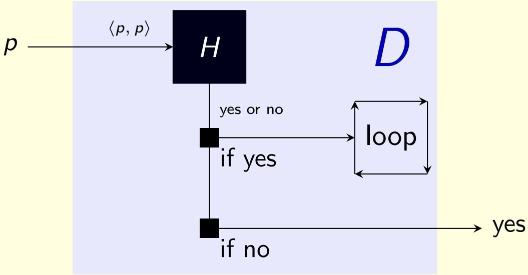
Theorem (Turing)

There is no Turing machine which accepts as input a TM p and input n for p and determines whether or not p with halt on n and produce an answer.

- Hard part! Turing showed that TMs are powerful enough to do computations involving other TMs. Indeed, he showed there is a universal machine which can simulate any TM.
- Other hard part! Turing's conceptual analysis to argue that his formalization correctly captures the intuitive notion of computability.
- Easy part! Do a diagonalization argument.

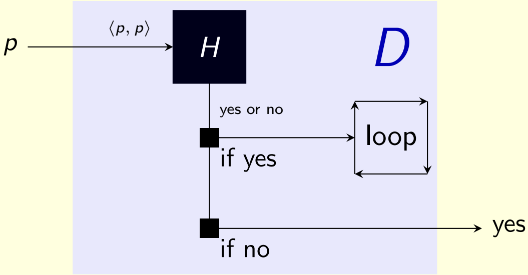
The easy part: the diagonalization argument

Toward a contradiction suppose H is a TM which decides whether or not p halts on input n. Let's build a new TM D.



The easy part: the diagonalization argument

Toward a contradiction suppose H is a TM which decides whether or not p halts on input n. Let's build a new TM D.



Now ask: what happens when D is input to D?
Then it halts iff it doesn't. ↯

From computability theory to proof theory

Let's talk about another kind of undecidability, in terms of what you can prove instead of what you can compute.

A very accurate and nuanced history of the incompleteness theorems

Find axioms that decide all questions of natural number arithmetic.

No.

The incompleteness theorems

Peano arithmetic (PA) axiomatizes natural number arithmetic: axioms of discretely ordered semirings + induction axioms.

Theorem (Gödel's first and second incompleteness theorems)

1. *No computably axiomatizable extension of* PA *is complete. There must be an arithmetic statement it neither proves nor disproves.*
2. PA *can neither prove nor disprove the consistency of* PA.

The incompleteness theorems

Peano arithmetic (PA) axiomatizes natural number arithmetic: axioms of discretely ordered semirings + induction axioms.

Theorem (Gödel's first and second incompleteness theorems)

1. *No computably axiomatizable extension of* PA *is complete. There must be an arithmetic statement it neither proves nor disproves.*
2. PA *can neither prove nor disprove the consistency of* PA.

- Hard part! (Arithmetization) Gödel showed that logical formulae can be coded as natural numbers, so statements about logic and proof can be coded as statements about natural numbers.
- Easy part! (Self-reference) Do a diagonalization argument.

Arithmetization

- Gödel's beta lemma states that arbitrary finite sequences can be coded as a single number, and this is provable within PA.
- Any finite mathematical object can be coded in arithmetic—e.g. logical formulae, Turing machines.
- So statements like "PA does not prove $0 = 1$" or "such and such Turing machine halts" can be cast as statements in arithmetic.

Arithmetization

0 substituted for $[s]_0$, and φ with $[s]_0+1$ substituted for $[s]_0$. Now for the gory details. We define the relation $\mathrm{PA}(x)$, expressing that x is the Gödel-number of a Peano axiom by the formula

$$x = n_1 \vee \cdots \vee x = n_{15} \vee$$
$$\exists y, s \subseteq x \exists n \leq s \left(\begin{array}{l} \mathrm{Form}(y) \wedge \mathrm{len}(s) = n \wedge \\ \forall i < \mathrm{len}(n)\, \mathrm{Free}(y, [s]_i) \wedge \forall j \leq y (\mathrm{Free}(y, j) \to \exists k \leq s\, [s]_k = j) \wedge \\ \exists t \subseteq s \exists u, w \left(\begin{array}{l} \mathrm{len}(t) = \mathrm{len}(s) - 1 \wedge \forall i < \mathrm{len}(t)\, [t]_i = [s]_{i+1} \wedge \\ u = \mathrm{Sub}(y, [s]_0, \ulcorner 0 \urcorner) \wedge w = \mathrm{Sub}(y, [s]_0, \ulcorner [s]_0 + 1 \urcorner) \wedge \\ x = \ulcorner (\forall t\, (u \wedge (\forall [s]_0\, (y \to w) \to \forall [s]_0\, y))) \urcorner \end{array} \right) \end{array} \right).$$

(Taken with permission from Victoria Gitman's lecture notes for Mathematical Logic, Spring 2013.)

Self-reference

Self-reference

Self-reference

- The Gödel fixed-point lemma states that a form of self-reference is possible for logical formulae.

  (Formally: for any formula $\varphi(x)$ there's a sentence σ so that σ is PA-provably equivalent to $\varphi(\sigma)$.)

Self-reference

- The Gödel fixed-point lemma states that a form of self-reference is possible for logical formulae.

  (Formally: for any formula $\varphi(x)$ there's a sentence σ so that σ is PA-provably equivalent to $\varphi(\sigma)$.)

- Also have self-reference for computability theory, via the Kleene recursion theorem.

  (Formally: for any partial computable function $F(x, y)$ there's a TM p so that p computes the function $y \mapsto F(p, y)$.)

Self-reference

- The Gödel fixed-point lemma states that a form of self-reference is possible for logical formulae.

  (Formally: for any formula $\varphi(x)$ there's a sentence σ so that σ is PA-provably equivalent to $\varphi(\sigma)$.)

- Also have self-reference for computability theory, via the Kleene recursion theorem.

  (Formally: for any partial computable function $F(x, y)$ there's a TM p so that p computes the function $y \mapsto F(p, y)$.)

A fun application: programming languages admit quines—programs that output their own source code.

```
;; Quine in Common Lisp
((lambda (x) (list x (list 'quote x)))
'(lambda (x) (list x (list 'quote x))))
```

Incompleteness and Turing machines

The incompleteness theorems can be recast as saying that whether certain Turing machines halt is undecidable.

A TM p:

- Look at all length 1 proofs from the first 1 axiom of PA.
- Then look at all length 2 proofs from the first 2 axioms of PA.
- $\vdots$
- If at any point you see a proof that ends with $0 = 1$, halt and output affirmatively.

Whether p halts is independent of PA.

Incompleteness and Turing machines

The incompleteness theorems can be recast as saying that whether certain Turing machines halt is undecidable.

A TM p:

- Look at all length 1 proofs from the first 1 axiom of PA.
- Then look at all length 2 proofs from the first 2 axioms of PA.
- $\vdots$
- If at any point you see a proof that ends with $0 = 1$, halt and output affirmatively.

Whether p halts is independent of PA.

Can also formulate in terms of what a TM enumerates.

- Do the same search through PA-proofs.
- But for each proof output the theorem then keep running.

Does this TM ever output $0 = 1$? PA does not prove one way or the other.

If you liked Gödel's incompleteness theorems, you'll love his completeness theorem

Theorem (Gödel's Completeness Theorem)

1. *A set of axioms T is consistent if and only if there is a structure T.*
2. *φ is true in every structure satisfying T if and only if φ is a theorem of T.*

(*Clarification: this is for axioms in first-order logic.*)

- This lets us move from talking about proofs, consistency, etc. to talking about structures. A lot of mathematicians—e.g. myself—find the latter perspective easier to think about!
- The incompleteness theorem plus the completeness theorem together imply there must be non-isomorphic structures satisfying the axioms of arithmetic.

If you liked Gödel's incompleteness theorems, you'll love his completeness theorem

Theorem (Gödel's Completeness Theorem)

1. *A set of axioms T is consistent if and only if there is a structure T.*
2. *φ is true in every structure satisfying T if and only if φ is a theorem of T.*

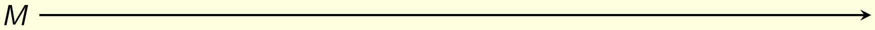
(Clarification: this is for axioms in first-order logic.)

- This lets us move from talking about proofs, consistency, etc. to talking about structures. A lot of mathematicians—e.g. myself—find the latter perspective easier to think about!
- The incompleteness theorem plus the completeness theorem together imply there must be non-isomorphic structures satisfying the axioms of arithmetic.

What could these even look like???

Nonstandard models of arithmetic

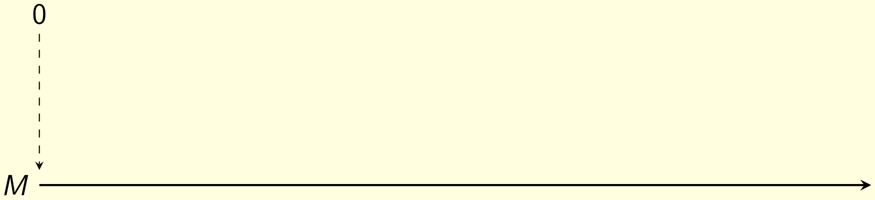
A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



- $X \subseteq M$ is definable if you can express $x \in X$ just by quantifying over the elements of M and using the ring operations and order of M.
- $X \subseteq M$ is inductive if $0 \in X$ and $a \in X \Rightarrow a + 1 \in X$ implies $X = M$.

Nonstandard models of arithmetic

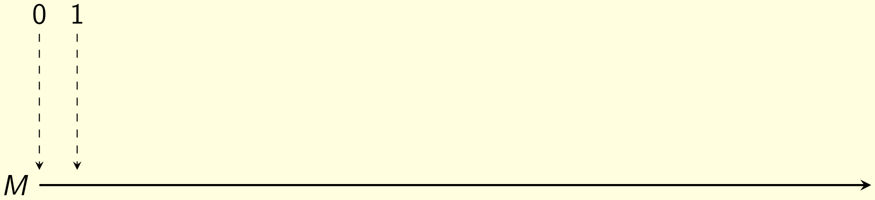
A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



M has a least element 0 because the set $\{x \in M : x \geq 0\}$ is inductive.

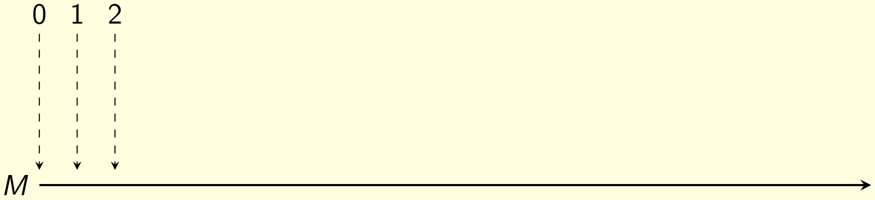
Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



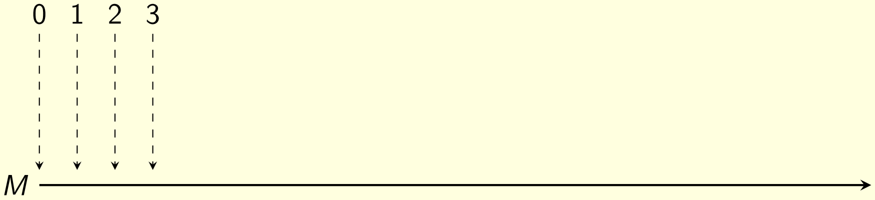
Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



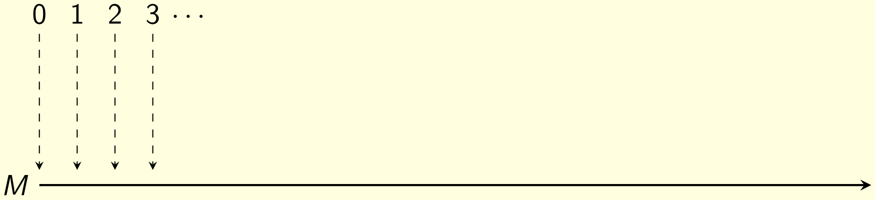
Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



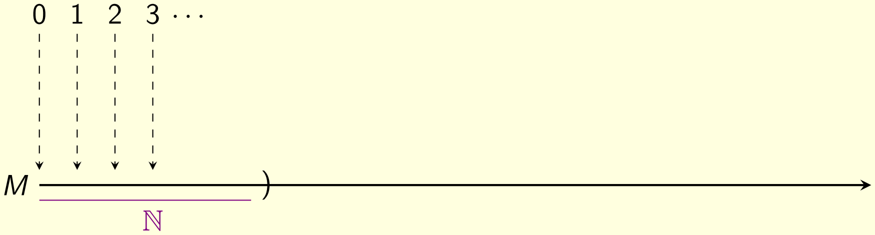
Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



Nonstandard models of arithmetic

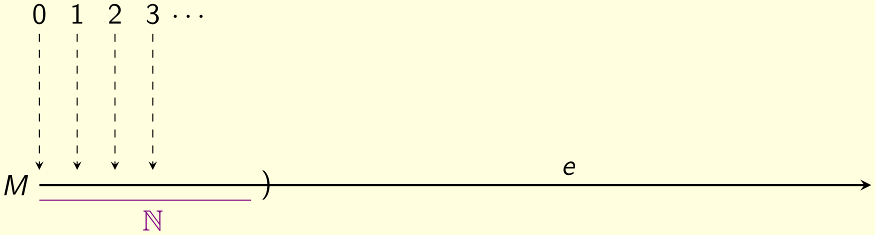
A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



$\mathbb{N}$ embeds as an initial segment on any model of arithmetic.

Nonstandard models of arithmetic

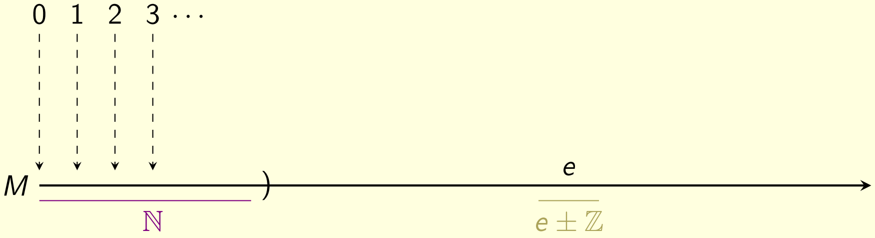
A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



If $e \in M \setminus \mathbb{N}$ then $e > n$ for all $n \in \mathbb{N}$.

Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



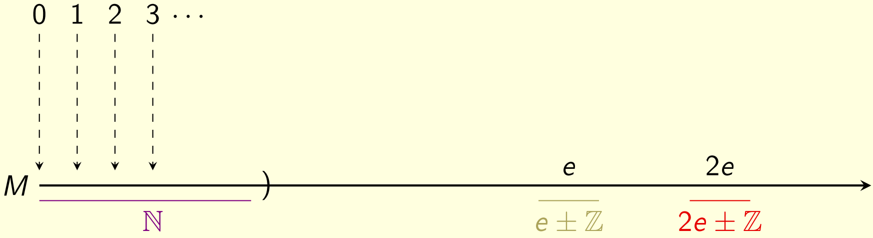
All non-zero elements have a predecessor because

$$\{0\} \cup \{a \in M : a \text{ has a predecessor}\}$$

satisfies the induction hypotheses.

Nonstandard models of arithmetic

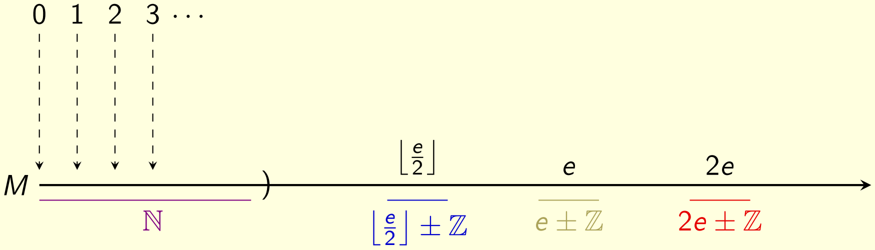
A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



$e + n < e + e = 2e$ for all $n \in \mathbb{N}$.

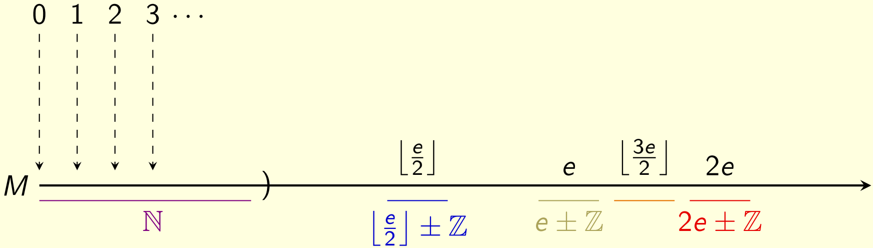
Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



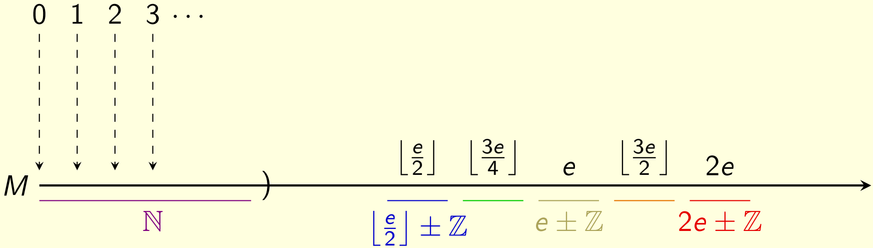
Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



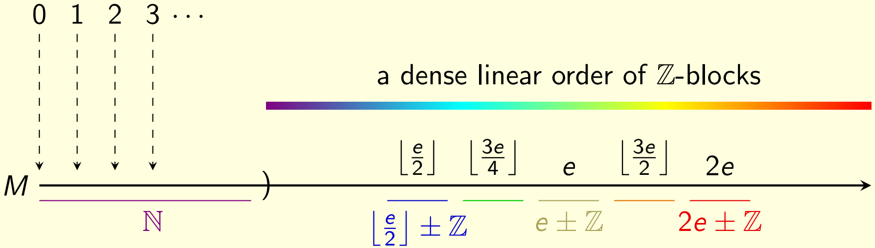
Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



Nonstandard models of arithmetic

A model of (Peano) arithmetic is a discretely ordered semiring whose definable subsets are inductive.



Nonstandard models of arithmetic

- First constructed by Thoralf Skolem. (Skolem used an ultrapower construction.)

Nonstandard models of arithmetic

- First constructed by Thoralf Skolem. (Skolem used an ultrapower construction.)
- If M is countable, then its ordertype is exactly $\mathbb{N} + \mathbb{Z} \cdot \mathbb{Q}$. (Because $\mathbb{Q}$ is the unique countable dense linear order without endpoints.)
- In particular, all countable nonstandard models of arithmetic are order-isomorphic.

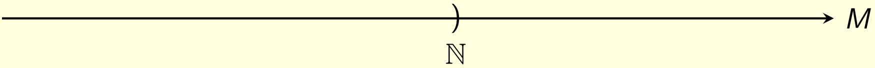
Nonstandard models of arithmetic

- First constructed by Thoralf Skolem. (Skolem used an ultrapower construction.)
- If M is countable, then its ordertype is exactly $\mathbb{N} + \mathbb{Z} \cdot \mathbb{Q}$. (Because $\mathbb{Q}$ is the unique countable dense linear order without endpoints.)
- In particular, all countable nonstandard models of arithmetic are order-isomorphic.
- Open Question (Harvey Friedman): $\mathbb{N}$ has the property that if a model of arithmetic is order-isomorphic to it then they are fully isomorphic. Does any other model of arithmetic have this property?

Nonstandard models of arithmetic

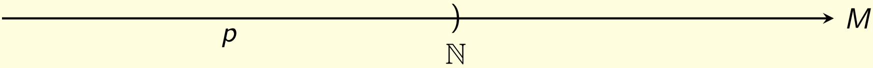
- First constructed by Thoralf Skolem. (Skolem used an ultrapower construction.)
- If M is countable, then its ordertype is exactly $\mathbb{N} + \mathbb{Z} \cdot \mathbb{Q}$. (Because $\mathbb{Q}$ is the unique countable dense linear order without endpoints.)
- In particular, all countable nonstandard models of arithmetic are order-isomorphic.
- Open Question (Harvey Friedman): $\mathbb{N}$ has the property that if a model of arithmetic is order-isomorphic to it then they are fully isomorphic. Does any other model of arithmetic have this property?
- (Stanley Tennenbaum) If M is nonstandard then neither the $+$ nor $\times$ of M is a computable function.

Turing machines in a nonstandard world



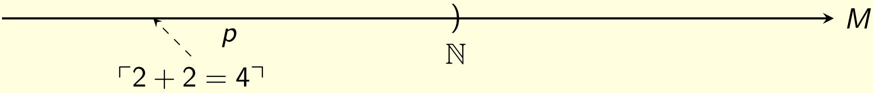
- Consider p the TM which enumerates the theorems of arithmetic.

Turing machines in a nonstandard world



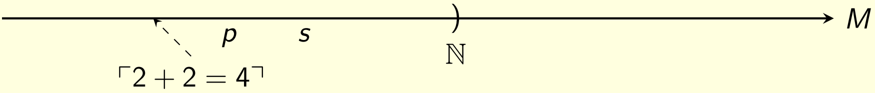
- Consider p the TM which enumerates the theorems of arithmetic.

Turing machines in a nonstandard world



- Consider p the TM which enumerates the theorems of arithmetic.

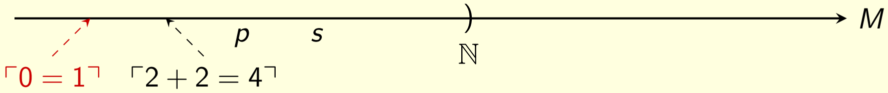
Turing machines in a nonstandard world



- Consider p the TM which enumerates the theorems of arithmetic.
- s is a computation log witnessing that p outputs $\ulcorner 2+2=4 \urcorner$.

  (s is a number coding the sequence of computation steps. Checking that s has this property only requires looking in $\mathbb{N}$.)

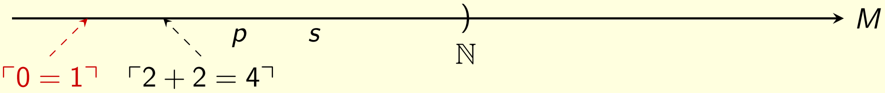
Turing machines in a nonstandard world



- Consider p the TM which enumerates the theorems of arithmetic.
- s is a computation log witnessing that p outputs $\ulcorner 2 + 2 = 4 \urcorner$.

  (s is a number coding the sequence of computation steps. Checking that s has this property only requires looking in $\mathbb{N}$.)
- If we run p in $\mathbb{N}$, then we never output $\ulcorner 0 = 1 \urcorner$.

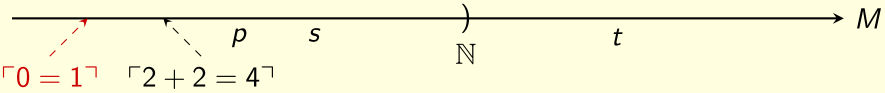
Turing machines in a nonstandard world



- Consider p the TM which enumerates the theorems of arithmetic.
- s is a computation log witnessing that p outputs $\ulcorner 2 + 2 = 4 \urcorner$.

  (s is a number coding the sequence of computation steps. Checking that s has this property only requires looking in $\mathbb{N}$.)
- If we run p in $\mathbb{N}$, then we never output $\ulcorner 0 = 1 \urcorner$.
- But what if we run p in nonstandard M which thinks arithmetic is inconsistent?

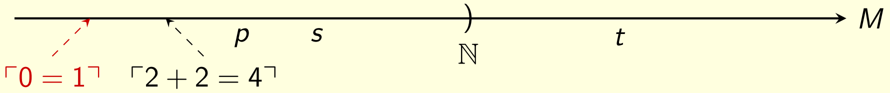
Turing machines in a nonstandard world



- Consider p the TM which enumerates the theorems of arithmetic.
- s is a computation log witnessing that p outputs $\ulcorner 2 + 2 = 4 \urcorner$.

  (s is a number coding the sequence of computation steps. Checking that s has this property only requires looking in $\mathbb{N}$.)
- If we run p in $\mathbb{N}$, then we never output $\ulcorner 0 = 1 \urcorner$.
- But what if we run p in nonstandard M which thinks arithmetic is inconsistent?
- Then there is a computation log t witnessing that p outputs $\ulcorner 0 = 1 \urcorner$. But t must be nonstandard!

Turing machines in a nonstandard world



- Consider p the TM which enumerates the theorems of arithmetic.
- s is a computation log witnessing that p outputs $\ulcorner 2+2=4 \urcorner$.

  (s is a number coding the sequence of computation steps. Checking that s has this property only requires looking in $\mathbb{N}$.)
- If we run p in $\mathbb{N}$, then we never output $\ulcorner 0 = 1 \urcorner$.
- But what if we run p in nonstandard M which thinks arithmetic is inconsistent?
- Then there is a computation log t witnessing that p outputs $\ulcorner 0 = 1 \urcorner$. But t must be nonstandard!
- By moving to a larger world we made p output more numbers.

The absoluteness of computability

In summary:

- The statement "the TM p outputs n for some input" is upward absolute—if it's true it stays true if we end-extend to a larger model.

  (Logicians call this sort of statement a Σ_1 statement. By the MRDP theorem, these are the statements equivalent to one whose only quantifiers are a block of $\exists$s.)

- But the statement "the TM p does not output n for some input" is not upward absolute. (It is downward absolute though.)

  (Logicians call this sort of statement a Π_1 statement.)

The absoluteness of computability

In summary:

- The statement "the TM p outputs n for some input" is upward absolute—if it's true it stays true if we end-extend to a larger model.

  (Logicians call this sort of statement a Σ_1 statement. By the MRDP theorem, these are the statements equivalent to one whose only quantifiers are a block of $\exists$s.)

  By Gödel's completeness theorem plus the last slide, Peano arithmetic proves every true (i.e. in $\mathbb{N}$) statement of this form.
- But the statement "the TM p does not output n for some input" is not upward absolute. (It is downward absolute though.)

  (Logicians call this sort of statement a Π_1 statement.)

  Both the first and second incompleteness theorems are about statements of this form.

Let's make a bet

- You'll think of a secret password (a finite sequence of natural numbers), and I'll try to guess it.
- But I'll tell you in advance the process I'll use to guess it, namely a specific a Turing machine.
- So you can use that info, if you like.
- If I successfully guess your number, you owe me a job with tenure :)

* (Adapted from a thought experiment in "A potential subtlety concerning the distinction between determinism and nondeterminism", W. Hugh Woodin, 2011.)

Let's make a bet

- You'll think of a secret password (a finite sequence of natural numbers), and I'll try to guess it.
- But I'll tell you in advance the process I'll use to guess it, namely a specific a Turing machine.
- So you can use that info, if you like.
- If I successfully guess your number, you owe me a job with tenure :)
- The trick: I'll wait and wait and wait for a very long time into the right nonstandard world, in which my TM will output your password.

* (Adapted from a thought experiment in "A potential subtlety concerning the distinction between determinism and nondeterminism", W. Hugh Woodin, 2011.)

Woodin's universal algorithm, first form

Theorem (Woodin)

There is a Turing machine p with the following properties.

1. *p provably enumerates a finite set.*
2. *Running p inside $\mathbb{N}$ never produces any output, i.e. it enumerates the empty set.*
3. *But, for any finite sequence s of natural numbers there is a nonstandard model of arithmetic M so that running p in M enumerates exactly s.*

Woodin's algorithm

(This construction for Woodin's theorem is due to Joel David Hamkins.)

The Turing machine p:

- p searches through the proofs of Peano arithmetic, looking at the theorems they prove.
- p is looking for a theorem of the form "p does **not** enumerate the sequence s", for s some nonempty sequence of numbers.

  (p can refer to itself by the Kleene Recursion theorem.)
- If p ever sees this, then p outputs the sequence s.

Woodin's algorithm

(This construction for Woodin's theorem is due to Joel David Hamkins.)

The Turing machine p:

- p searches through the proofs of Peano arithmetic, looking at the theorems they prove.
- p is looking for a theorem of the form "p does **not** enumerate the sequence s", for s some nonempty sequence of numbers.

  (p can refer to itself by the Kleene Recursion theorem.)
- If p ever sees this, then p outputs the sequence s.

Claim: Run in $\mathbb{N}$, p outputs the empty sequence.

Otherwise p outputs some s. So Peano arithmetic proves this true Σ_1 statement. But by the definition of p, this also means that Peano arithmetic proves that p does not output s. This would mean that Peano arithmetic is inconsistent. But it's not.

Checking the extension property

Definition (The Turing machine p)

- p searches through the proofs of Peano arithmetic, looking for a theorem of the form "p does **not** enumerate the sequence s", for s some nonempty sequence of numbers.
- If p ever sees this, then p outputs the sequence s.

Fix a finite sequence of natural numbers s. We want to find a nonstandard model of arithmetic M in which running p outputs s.

Checking the extension property

Definition (The Turing machine p)

- p searches through the proofs of Peano arithmetic, looking for a theorem of the form "p does **not** enumerate the sequence s", for s some nonempty sequence of numbers.
- If p ever sees this, then p outputs the sequence s.

Fix a finite sequence of natural numbers s. We want to find a nonstandard model of arithmetic M in which running p outputs s.

Claim: Peano arithmetic + "p outputs s" is consistent.

Checking the extension property

Definition (The Turing machine p)

- p searches through the proofs of Peano arithmetic, looking for a theorem of the form "p does **not** enumerate the sequence s", for s some nonempty sequence of numbers.
- If p ever sees this, then p outputs the sequence s.

Fix a finite sequence of natural numbers s. We want to find a nonstandard model of arithmetic M in which running p outputs s.

Claim: Peano arithmetic + "p outputs s" is consistent.

Otherwise "p does not output s" is a theorem of Peano arithmetic. But then running p in $\mathbb{N}$ would output a nonempty sequence. We just saw that is not the case.

Checking the extension property

Definition (The Turing machine p)

- p searches through the proofs of Peano arithmetic, looking for a theorem of the form "p does **not** enumerate the sequence s", for s some nonempty sequence of numbers.
- If p ever sees this, then p outputs the sequence s.

Fix a finite sequence of natural numbers s. We want to find a nonstandard model of arithmetic M in which running p outputs s.

Claim: Peano arithmetic + "p outputs s" is consistent.

Otherwise "p does not output s" is a theorem of Peano arithmetic. But then running p in $\mathbb{N}$ would output a nonempty sequence. We just saw that is not the case.

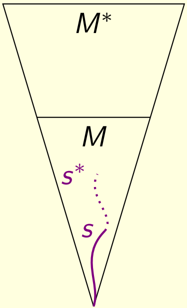
So by Gödel's completeness theorem we can find a model of arithmetic in which p outputs s. □

Woodin's universal algorithm, general form

Theorem (Woodin)

There is a Turing machine p with the following properties.

1. *p provably enumerates a finite set.*
2. *Running p inside $\mathbb{N}$ never produces any output, i.e. it enumerates the empty set.*
3. *Suppose M a nonstandard model of arithmetic in which p enumerates s and that s^* is a sequence in M which extends s. Then we can further end-extend M to a larger model of arithmetic M^* in which p enumerates s^*.*

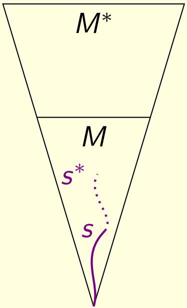


Woodin's universal algorithm, general form

Theorem (Woodin)

There is a Turing machine p with the following properties.

1. *p provably enumerates a finite set.*
2. *Running p inside $\mathbb{N}$ never produces any output, i.e. it enumerates the empty set.*
3. *Suppose M a nonstandard model of arithmetic in which p enumerates s and that s^* is a sequence in M which extends s. Then we can further end-extend M to a larger model of arithmetic M^* in which p enumerates s^*.*



Proof idea: Do a similar argument, but internally to M. Need some more technical lemmata to check that the argument can be arithmetized.

What about the fourth pillar of mathematical logic?

Traditionally mathematical logic has been divided into four pillars: computability theory, proof theory, model theory, and set theory.*

So far in this talk we've seen the first three of these. What about the fourth?

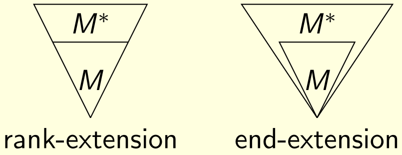
* (Recent work e.g. in categorical logic undermines this taxonomy.)

What about the fourth pillar of mathematical logic?

Traditionally mathematical logic has been divided into four pillars: computability theory, proof theory, model theory, and set theory.*

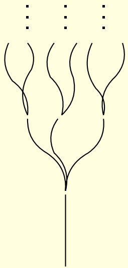
So far in this talk we've seen the first three of these. What about the fourth?

Analogous to Woodin's universal algorithm in arithmetic there are results in set theory, where there's more than one sensible notion of extension to consider. In set theory: Hamkins and Woodin construct a universal sequence for rank-extensions, and Hamkins and Williams construct a universal sequence for end-extensions.



* (Recent work e.g. in categorical logic undermines this taxonomy.)

Arithmetic potentialism



- Imagine climbing through the tree of nonstandard models of arithmetic, continually end-extending.
- This potentialist system gives a nonstandard twist on Aristotle's notion of the potential infinite.
- There is a natural interpretation in modal logic—extend ordinary logic by adding two new operators
  - $\Box\varphi$ means φ is necessarily true—true in all extensions.
  - $\Diamond\varphi$ means φ is possibly true—true in some extension.
- (Hamkins) Can use Woodin's universal algorithm to calculate which modal assertions are valid (true in any world under any substitution of variables).
- (Hamkins) There are models of arithmetic which satisfy the maximality principle—if $\Diamond\Box\varphi$ then φ.

Thank you for listening!