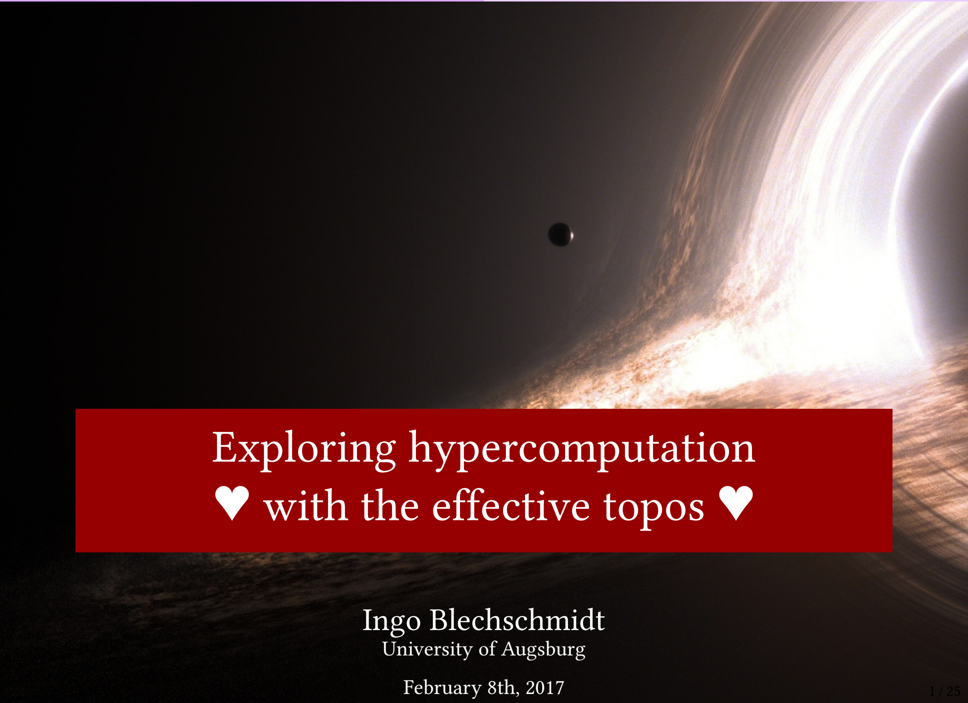


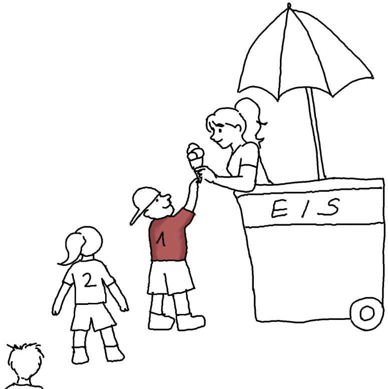
Exploring hypercomputation ♥ with the effective topos ♥

Ingo Blechschmidt
University of Augsburg

February 8th, 2017

Part I

A crash course on ordinal numbers

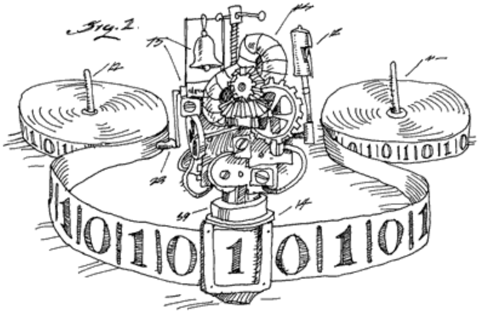


The Wikipedia article on ordinal numbers is a good starting point. For the purposes of this talk, it suffices to have some intuitive grasp of the ordinal number line:

$$0,\ 1,\ 2,\ \ldots,\ \omega,\ \omega+1,\ \omega+2,\ \ldots,\ \omega\cdot 2,\ \omega\cdot 2+1,\ \ldots$$

Part II

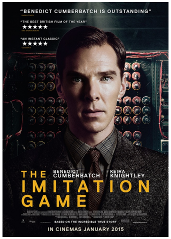
(Super) Turing machines



Basics on Turing machines

- Turing machines are idealised computers operating on an **infinite tape** according to a **finite list** of rules.
- The concept is astoundingly robust.
- A subset of $\mathbb{N}$ is **enumerable by a Turing machine** if and only if it's a Σ_1-set.

Alan Turing
(* 1912, † 1954)


"BENEDICT CUMBERBATCH IS OUTSTANDING"
"THE BEST BRITISH FILM OF THE YEAR"
★★★★★
"AN INSTANT CLASSIC"
★★★★★
BENEDICT CUMBERBATCH
KEIRA KNIGHTLEY
THE IMITATION GAME
BASED ON THE INCREDIBLE TRUE STORY
IN CINEMAS JANUARY 2015
worth watching

Alison Bechdel
(* 1960)

- There are explicit examples for Turing machines with a very small number of states for which their halting behaviour is unknown. There are other examples for which their halting behaviour is independent of standard Zermelo–Fraenkel set theory.

- Besides Turing machines, there are alternative models for computation, for instance lambda calculus and register machines. For functions $\mathbb{N} \to \mathbb{N}$, these models all yield the same notion of computabitility. For higher-order functions, where the domain is a set of functions, the models typically differ in which functions they deem computable.

- A set $M \subseteq \mathbb{N}$ is *enumerable by a Turing machines* (or *recursively enumerable*) if and only if there is a Turing machine which outputs all the elements of M (in an arbitrary order, using some fixed encoding).

- A set $M \subseteq \mathbb{N}$ is a Σ_1-set if and only if there is a Σ_1-formula φ of first-order arithmetic containing exactly one free variable such that $M = \{n \in \mathbb{N} \mid \varphi(n)\}$. A formula φ is a Σ_1-formula if and only if it is of the form

$$\exists m_1.\ \exists m_2.\ \ldots \exists m_k.\ \heartsuit,$$

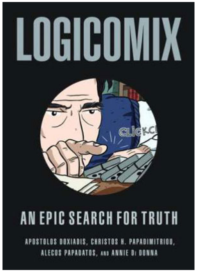
  where the existential quantifiers range over the natural numbers and no further unbounded quantifiers (of any kind) may appear in the subformula $\heartsuit$.

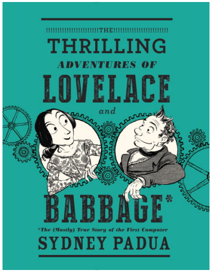
There is also a deep link between computability and number theory: A set $M \subseteq \mathbb{N}$ is enumerable by a Turing machine if and only if it is a *diophantine set*, that is of the form

$$M = \{n \in \mathbb{N} \mid \text{the eq. } f(n, x_1, \ldots, x_m) = 0 \text{ has a solution}\},$$

where f is a polynomial with integral coefficients and “solution” means integral solution.

Just as *Logicomix: An Epic Search for Truth* illustrates the foundational quest in mathematics (starring Cantor, Hilbert, Gödel, Turing, and others), the graphic novel *The Thrilling Adventures of Lovelace and Babbage* explores the history of programmable computers.


LOGICOMIX
CLICK
AN EPIC SEARCH FOR TRUTH
APOSTOLOS DOXIADIS, CHRISTOS H. PAPADIMITRIOU,
ALECOS PAPADATOS, AND ANNIE DI DONNA



Super Turing machines

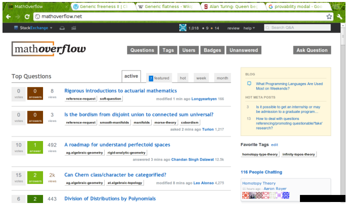
With super Turing machines, the time axis is more interesting:

- normal: $0, \ 1, \ 2, \ \ldots$
- super: $\ 0, \ 1, \ 2, \ \ldots, \ \omega, \ \omega + 1, \ \ldots, \ \omega \cdot 2, \ \omega \cdot 2 + 1, \ \ldots\ldots\ldots$

On reaching a limit ordinal time step like ω or $\omega \cdot 2$,

- the machine is put into a designated state,
- the read/write head is moved to the start of the tape, and
- the tape is set to the “lim sup” of all its previous contents.

Joel David Hamkins



Top Questions

0 votes 0 answers 8 views Rigorous introductions to actuarial mathematics
reference-request soft-question modified 1 min ago Longyearbyen 166

0 votes 0 answers 3 views Is the bordism from disjoint union to connected sum universal?
reference-request smooth-manifolds manifolds morse-theory cobordism asked 2 mins ago Turion 1,217

10 votes 1 answer 492 views A roadmap for understand perfectoid spaces
ag.algebraic-geometry rigid-analytic-geometry answered 3 mins ago Chandan Singh Dalawat 12.5k

15 votes 2 answers 2k views Can Chern class/character be categorified?
ag.algebraic-geometry at.algebraic-topology modified 8 mins ago Leo Alonso 4,275

6 2 443 Division of Distributions by Polynomials

BLOG
What Programming Languages Are Used Most on Weekends?

HOT META POSTS
3 Is it possible to get an internship or may be admission to a graduate program...
13 How to deal with questions referencing/promoting questionable/'fake' research?

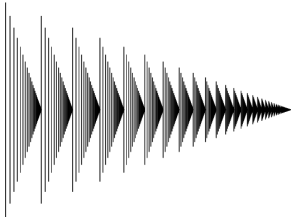
Favorite Tags edit
homotopy-type-theory infinity-topos-theory

116 People Chatting
Homotopy Theory
11 hours ago - Aaron Royer

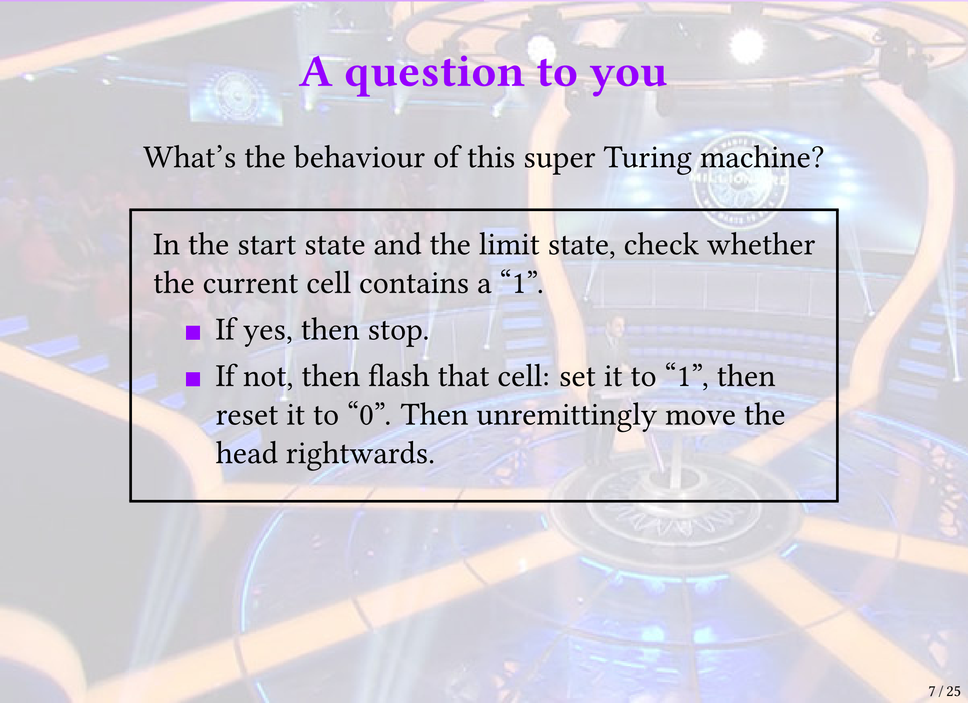
MathOverflow

Andy Lewis

We can imagine a super Turing machine to take one second for its first computational step, half a second for the next step, a quarter of a second for the step after that, and so on. After two seconds, the machine will have completed infinitely many steps.



The fine print: This visual illustration only works for those ordinals which can be embedded into the ordinary real number line $\mathbb{R}$. These are exactly the countable ordinals, those ordinals for which the set of predecessors is countable. Luckily, the interesting behaviour of a super Turing machine always takes place in the realm of countable ordinal numbers (even though this is not a requirement we put on super Turing machines a priori).

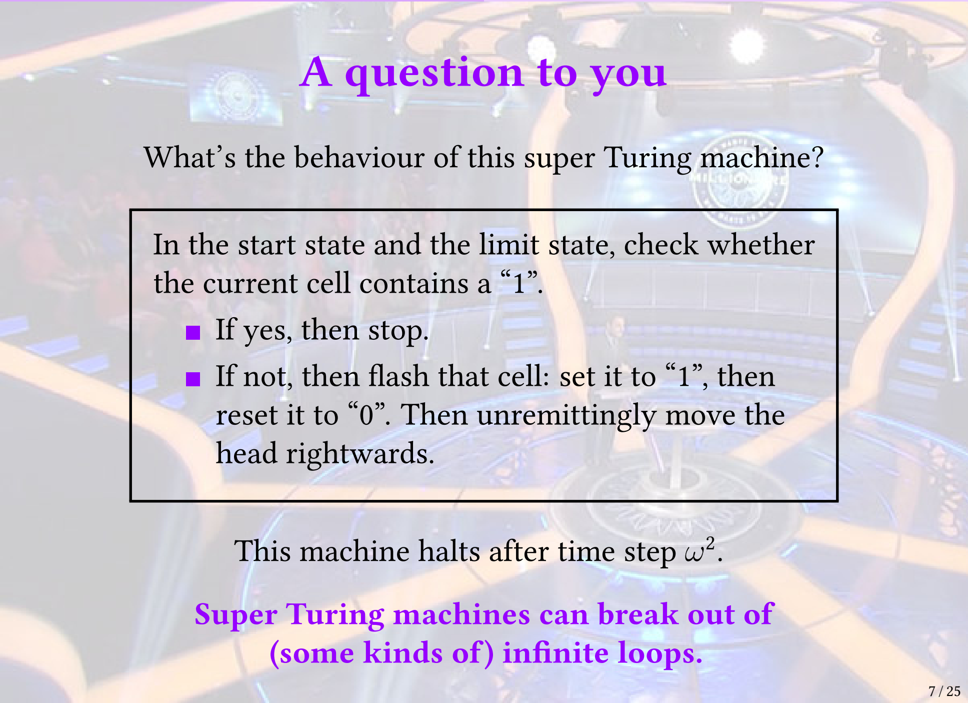


A question to you

What's the behaviour of this super Turing machine?

In the start state and the limit state, check whether the current cell contains a "1".

- If yes, then stop.
- If not, then flash that cell: set it to "1", then reset it to "0". Then unremittingly move the head rightwards.



A question to you

What's the behaviour of this super Turing machine?

In the start state and the limit state, check whether the current cell contains a "1".

- If yes, then stop.
- If not, then flash that cell: set it to "1", then reset it to "0". Then unremittingly move the head rightwards.

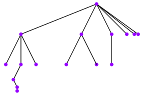
This machine halts after time step ω^2.

Super Turing machines can break out of (some kinds of) infinite loops.

There is an ASCII video of a run of that machine.

What can super Turing machines do?

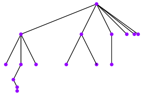
- Everything ordinary Turing machines can do.
- Verify number-theoretic statements.
- Decide whether a given ordinary Turing machine halts.
- Simulate super Turing machines.
- Decide Π^1_1- and Σ^1_1-statements:
  - “For any function $\mathbb{N} \to \mathbb{N}$ it holds that …”
  - “There is a function $\mathbb{N} \to \mathbb{N}$ such that …”

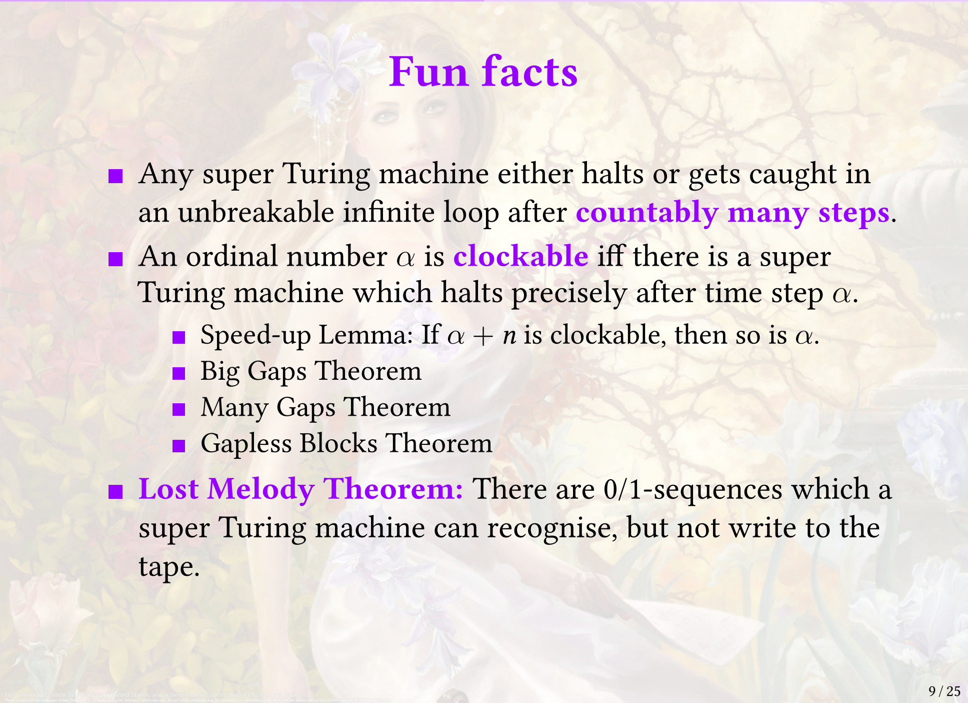


What can super Turing machines do?

- Everything ordinary Turing machines can do.
- Verify number-theoretic statements.
- Decide whether a given ordinary Turing machine halts.
- Simulate super Turing machines.
- Decide Π^1_1- and Σ^1_1-statements:
  - "For any function $\mathbb{N} \to \mathbb{N}$ it holds that …"
  - "There is a function $\mathbb{N} \to \mathbb{N}$ such that …"

But: Super Turing machines can't compute all functions and can't write arbitrary 0/1-sequences to the tape.





Fun facts

- Any super Turing machine either halts or gets caught in an unbreakable infinite loop after **countably many steps**.
- An ordinal number α is **clockable** iff there is a super Turing machine which halts precisely after time step α.
  - Speed-up Lemma: If $\alpha + n$ is clockable, then so is α.
  - Big Gaps Theorem
  - Many Gaps Theorem
  - Gapless Blocks Theorem
- **Lost Melody Theorem:** There are 0/1-sequences which a super Turing machine can recognise, but not write to the tape.

- In view of the fact that super Turing machines always halt at countable ordinal numbers (if they halt at all), the ability of super Turing machines to decide Π^1_1- and Σ^1_1-statements is even more astounding.
- That there are at least some ordinal numbers which are not clockable, can be seen by a simple cardinality argument: There are only countably many super Turing machines, but a proper class worth of ordinal numbers.
- The Big Gaps Theorem states: For any clockable ordinal α there is a gap of length $\geq \alpha$ in the clockable ordinals.

- The seminal paper introducing super Turing machines is a joy to read. It contains the statements and proofs of all the cited theorems.
- I’m using the adjectives “super”, “hyper”, and “infinite-time” interchangably.

Part III

The effective topos

For any model $\mathcal{M}$ of computation, such as Turing machines (TM), super Turing machines (STM), or lambda calculus (λC), there is an associated *effective topos* $\mathrm{Eff}(\mathcal{M})$. Formally, a topos is a certain kind of category; intuitively, a topos is an alternate mathematical universe in which the usual laws of logic may not necessarily hold.

The standard universe, in which most of mathematics happens, is the topos Set.

One can even employ machines of the real physical world as the "model" $\mathcal{M}$ used for constructing the effective topos (RW). Statements about the resulting topos are then statements about the real world instead of formal mathematical statements.

The effective topos

- "$1 + 1 = 2$."
- "Any number is either prime or not."
- "Any function $\mathbb{N} \to \mathbb{N}$ is either the zero function or not."
- "Any function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."
- "Any function $\mathbb{R} \to \mathbb{R}$ is continuous."

The effective topos

- "$1 + 1 = 2$."
  True in Set, true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any number is either prime or not."
- "Any function $\mathbb{N} \to \mathbb{N}$ is either the zero function or not."
- "Any function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."
- "Any function $\mathbb{R} \to \mathbb{R}$ is continuous."

The effective topos

- "$1 + 1 = 2$."
  True in Set, true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any number is either prime or not."
  Trivially true in Set, nontrivially true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any function $\mathbb{N} \to \mathbb{N}$ is either the zero function or not."
- "Any function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."
- "Any function $\mathbb{R} \to \mathbb{R}$ is continuous."

The effective topos

- "$1 + 1 = 2$."
  True in Set, true in Eff(TM).
- "Any number is either prime or not."
  Trivially true in Set, nontrivially true in Eff(TM).
- "Any function $\mathbb{N} \to \mathbb{N}$ is either the zero function or not."
  Trivially true in Set, false in Eff(TM).
- "Any function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."
- "Any function $\mathbb{R} \to \mathbb{R}$ is continuous."

The effective topos

- "$1 + 1 = 2$."
  True in Set, true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any number is either prime or not."
  Trivially true in Set, nontrivially true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any function $\mathbb{N} \to \mathbb{N}$ is either the zero function or not."
  Trivially true in Set, false in $\mathrm{Eff}(\mathrm{TM})$.
- "Any function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."
  False in Set, trivially true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any function $\mathbb{R} \to \mathbb{R}$ is continuous."

The effective topos

- "$1 + 1 = 2$."
  True in Set, true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any number is either prime or not."
  Trivially true in Set, nontrivially true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any function $\mathbb{N} \to \mathbb{N}$ is either the zero function or not."
  Trivially true in Set, false in $\mathrm{Eff}(\mathrm{TM})$.
- "Any function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."
  False in Set, trivially true in $\mathrm{Eff}(\mathrm{TM})$.
- "Any function $\mathbb{R} \to \mathbb{R}$ is continuous."
  False in Set, nontrivially true in $\mathrm{Eff}(\mathrm{TM})$.

First steps in the effective topos

$\mathrm{Eff}(\mathrm{TM}) \models$ "For any number n there is a prime $p > n$."

means:

There is a Turing machine which reads a number n as input and outputs a prime number $p > n$.

"Realisability Theory"

First steps in the effective topos

$\mathrm{Eff}(\mathrm{TM}) \models$ "For any number n there is a prime $p > n$."
means:
There is a Turing machine which reads a number n as input and outputs a prime number $p > n$.

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any number has a prime factor decomposition."
means:
There is a Turing machine which reads a number n as input and outputs a list of primes, the product of which is n.

First steps in the effective topos

$\mathrm{Eff}(\mathrm{TM}) \models$ "For any number n there is a prime $p > n$."

means:

There is a Turing machine which reads a number n as input and outputs a prime number $p > n$.

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any number has a prime factor decomposition."

means:

There is a Turing machine which reads a number n as input and outputs a list of primes, the product of which is n.

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any number is either prime or not prime."

means:

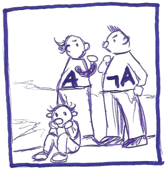
There is a Turing machine which reads a number n as input and outputs YES or NO depending on whether n is prime or not.

What's true in alternate toposes?

Metatheorem: If a statement has a **constructive proof**, then it holds in **any topos**.

Constructive logic is like classical logic, except we don't suppose the **law of excluded middle** (LEM), which says:

- "Any statement is either true or not true."
- "If a statement is *not not* true, then it's true."



- The technical term for “constructive logic” is “intuitionistic logic”. “Constructive logic” or even “constructive mathematics” is more of an umbrella term which might mean any of several related, but not equivalent systems.

- The law of excluded middle is the axiom which allows us to argue by contradiction. But the word “contradiction” is still allowed in constructive mathematics; see below.

- The proof of the metatheorem is constructive.

Nonconstructive proofs

Theorem. There are **irrational** numbers x und y such that x^y is rational.

Nonconstructive proofs

Theorem. There are **irrational** numbers x und y such that x^y is rational.

Proof. Either $\sqrt{2}^{\sqrt{2}}$ is rational or not.

1. In the first case we are done.
2. In the second case we set $x := \sqrt{2}^{\sqrt{2}}$ and $y := \sqrt{2}$.
   Then $x^y = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = \sqrt{2}^2 = 2$ is rational.

The proof is nice and short. However, after having seen the proof, we are still not able to give an example of irrational numbers x, y such that x^y is rational! The proof was *non-constructive.* If we want to extract explicit witnesses from the proof, the proof has to be constructive, such as this one:

> *Set* $x := \sqrt{2}$ *and* $y := \log_{\sqrt{2}} 3$*. Then* $x^y = 3$ *is rational. The proof that* y *is irrational is even easier than the proof that* $\sqrt{2}$ *is irrational.*

It turns out that from all the axioms of classical logic, exactly one is responsible for non-constructivity: the law of excluded middle.

Appreciating constructive logic

At first sight, dropping the law of excluded middle looks like a sad thing to do. It's a useful axiom! However:

- The axiom is not needed as often as one would think.
- The abstinence is good for your mental hygiene.
- Constructive logic allows for finer distinctions.
- From constructive proofs one can mechanically extract programs which witness the proved statements.
- **Dropping the law of excluded middle allows to add curious unconvential axioms.**

We're used to immediately cancel double negations. Since that's constructively not possible, we have to exercise a bit of linguistic caution:

- *Proof of a negated statement (constructively acceptable):* We want to verify $\neg\psi$. Assume that ψ holds. Then …, that's a contradiction. Therefore $\neg\psi$.

  This is a constructively valid argument, since in constructive (and classical) logic $\neg\psi$ is defined as the implication $(\psi \Rightarrow \bot)$, where $\bot$ is the formal logical symbol for falsity, a contradiction.

- *Proof by contradiction (constructively not acceptable a priori):* We want to verify φ. Assume that φ is false. Then …, that's a contradiction. Therefore φ holds.

  The constructively valid part of this argument only shows that φ does *not not* hold: $\neg\neg\varphi$.

Several years ago a video showing Kate Moss consuming drugs surfaced. From the video it was clear that the drugs were either of some type A or of some type B, but there was no direct evidence for either type. Kate Moss was not prosecuted; in this sense, the judicial system operated on constructive logic. Check out Dan Piponi's blog post about this topic.

Constructive logic allows for finer distictions than classical logic. For instance, if we know that the key to our apartment has to be somewhere in the apartment (since we used it to enter last night) but we can't find it right now, we can constructively only justify

$$\neg\neg(\exists x.\ \text{the key is at position } x),$$

not the stronger statement

$$\exists x.\ \text{the key is at position } x,$$

because for justifying the stronger statement we'd had to give an explicit witness of the existential statement (the position of the key).

(Both examples do not quite work, since constructive mathematics is (like classical mathematics) indifferent to our personal state of knowledge.)

Constructive mathematicians do *not* claim that the law of excluded middle is false (that is, that its negation holds). In fact, some instances of the law of excluded middle are true intuitionistically: For example one can show by induction that any natural number is zero or is not zero. Constructive mathematicians simply don't suppose that the law of excluded holds generally.

The analogous statement about real numbers does not have a constructive proof. This is linked with a familiar fact from programming: It's unbeseeming to compare floating point numbers for equality, while there is no such problem with comparing integers.

Philosophers don't only study what's *true*, but also what *should be true*, what's *possible*, what's *necessary*, what somebody *knows* or somebody *believes*. This is formalised using *modal operators*.

Mathematicians don't need to envy that greater scope: In constructive mathematics there is too a multitude of modal operators. Double negation is the most prominent example.

Andrej Bauer gave a very insightful talk about the merits of constructive mathematics at the Institute for Advanced Study. His talk is a joy to watch!

- video
- accompanying article

LEM for equality of functions

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any function $f : \mathbb{N} \to \mathbb{N}$ is either the zero function or not."

means:

There is a Turing machine which reads the source of a Turing machine M, which computes a function $\mathbb{N} \to \mathbb{N}$, as input, and finds out whether M always yields zero or not.

That's false.

LEM for equality of functions

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any function $f : \mathbb{N} \to \mathbb{N}$ is either the zero function or not."

means:

There is a Turing machine which reads the source of a Turing machine M, which computes a function $\mathbb{N} \to \mathbb{N}$, as input, and finds out whether M always yields zero or not.

That's false.

The statement is true in $\mathrm{Eff}(\mathrm{STM})$, the effective topos associated to super Turing machines.

LEM for the halting problem

$\mathrm{Eff}(\mathrm{TM}) \models$ “Any Turing machine halts or doesn’t halt.”

means:

There is a Turing machine which reads the source of a Turing machine M as input and finds out whether M halts or not.

That’s false.

LEM for the halting problem

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any Turing machine halts or doesn't halt."

means:

There is a Turing machine which reads the source of a Turing machine M as input and finds out whether M halts or not.

That's false.

The statement is true in $\mathrm{Eff}(\mathrm{STM})$.

LEM for equality of real numbers

The statement

“Every real number is either zero or not zero.”

is in constructive logic equivalent to

“Every Turing machine halts or doesn’t halt.”,

so it’s not true in $\mathrm{Eff}(\mathrm{TM})$, but in $\mathrm{Eff}(\mathrm{STM})$.

LEM for equality of real numbers

The statement

“Every real number is either zero or not zero.”

is in constructive logic equivalent to

“Every Turing machine halts or doesn’t halt.”,

so it’s not true in $\mathrm{Eff}(\mathrm{TM})$, but in $\mathrm{Eff}(\mathrm{STM})$.

For a Turing machine M consider the real number $0.000\ldots$ whose n’th decimal digit is a one iff M halts after step n.

For a real number x consider the Turing machine which searches the digits of x for a nonzero digit.

Is the statement true in Eff(RW), the effective topos associated to machines of the real world? That means: Is it possible to build a halting oracle in the real world? A machine which reads the description of a Turing machine and then outputs whether the Turing machine halts or not?

Since this is not about the halting problem for machines of the real world (which is not decidable by machines of the real world, by the usual argument), but only about the halting problem for Turing machines, the answer *might* be "yes". Tricks using black holes and relativistic time dilation might be possible.

More details on Eff(RW) are contained in the very accessible book chapter Intuitionistic Mathematics and Realizability in the Physical World by Andrej Bauer.

Markov's principle

$\mathrm{Eff}(\mathrm{TM}) \models$ "For any function $f : \mathbb{N} \to \mathbb{N}$ which is not the zero function, there is a number $n \in \mathbb{N}$ such that $f(n) \neq 0$."

means:

> There is a Turing machine which reads the source of a Turing machine M, which computes a function $\mathbb{N} \to \mathbb{N}$ which is not the zero function, as input and outputs a number n such that $M(n)$ is not zero.

Markov's principle

$\mathrm{Eff}(\mathrm{TM}) \models$ "For any function $f : \mathbb{N} \to \mathbb{N}$ which is not the zero function, there is a number $n \in \mathbb{N}$ such that $f(n) \neq 0$."

means:

There is a Turing machine which reads the source of a Turing machine M, which computes a function $\mathbb{N} \to \mathbb{N}$ which is not the zero function, as input and outputs a number n such that $M(n)$ is not zero.

That's true! By unbounded search.

The axiom of choice

A set X ist **projective** if and only if, for any set Y and any formula $\varphi(x, y)$ with parameters $x \in X$, $y \in Y$ it holds that:

> If for any $x \in X$ there is an element $y \in Y$ such that $\varphi(x, y)$, then there is a map $f : X \to Y$ such that $\varphi(x, f(x))$ for all $x \in X$.

$$\forall x \in X.\ \exists y \in Y.\ \varphi(x, y) \implies \exists f \in Y^X.\ \forall x \in X.\ \varphi(x, f(x))$$

The **axiom of choice** of classical mathematics states that any set is projective.

- In Eff(TM) the set $\mathbb{N}$ is projective, but $\mathbb{N}^{\mathbb{N}}$ is not.
- In Eff(STM) the sets $\mathbb{N}$ and $\mathbb{N}^{\mathbb{N}}$ are projective.
- In Eff(RW) the set $\mathbb{N}^{\mathbb{N}}$ is projective if black boxes are possible.

Finite sets are projective even in constructive mathematics.

By $\mathbb{N}^{\mathbb{N}}$, we mean the set of all functions $\mathbb{N} \to \mathbb{N}$.

A good way to convince oneself that the axiom of choice is indeed a nontrivial statement is to look at what the statement "$\mathbb{N}^\mathbb{N}$ is projective" means in Eff(TM). It means that the statement

> There is a Turing machine M which reads the description of a Turing machine P, which computes a function $f : \mathbb{N} \to \mathbb{N}$, as input and outputs an element $M(P) \in Y$ together with a witness of $\varphi(f, M(P))$.

implies

> There is a computable map g from the set of computable maps $f : \mathbb{N} \to \mathbb{N}$ to the set Y such that for all such functions f the statement $\varphi(f, g(f))$ holds in Eff(TM).

On first sight, this implications seems to be true: It seems that one could define the looked-for map g to be simply the map computed by M. But that doesn't necessarily work: If $f : \mathbb{N} \to \mathbb{N}$ is a map which is computed by a Turing machine P, then the element $M(P)$ might depend on the concrete realisation of f by the machine P. Different implementations of f by different Turing machines P' might yield different elements $M(P')$. Therefore the resulting "map" g might be multi-valued, so not a proper map.

Searching uncountable sets

"For any function $f : \mathbb{N} \to \mathbb{B}$ from numbers to $\mathbb{B} = \{0, 1\}$ there either exists a number n such that $f(n) = 1$ or there is no such number."

This statement is false in $\mathrm{Eff}(\mathrm{TM})$.

"For any function $P : \mathbb{B}^{\mathbb{N}} \to \mathbb{B}$ from infinite lists of booleans to booleans, there either exists a list x such that $P(x) = 1$ or there is no such list."

Searching uncountable sets

> “For any function $f : \mathbb{N} \to \mathbb{B}$ from numbers to $\mathbb{B} = \{0, 1\}$ there either exists a number n such that $f(n) = 1$ or there is no such number.”

This statement is false in $\mathrm{Eff}(\mathrm{TM})$.

> “For any function $P : \mathbb{B}^{\mathbb{N}} \to \mathbb{B}$ from infinite lists of booleans to booleans, there either exists a list x such that $P(x) = 1$ or there is no such list.”

This statement is true in $\mathrm{Eff}(\mathrm{TM})$!

It's somewhat miraculous that the countable set $\mathbb{N}$ can not be searched, but the uncountable set

$$\mathbb{B}^{\mathbb{N}} = \prod_{n=0}^{\infty} \mathbb{B} = \{(x_0, x_1, \ldots)\}$$

can. There is a deeper topological reason for this fact: The space $\mathbb{B}^{\mathbb{N}}$ is compact (by Tychonoff's theorem) while $\mathbb{N}$ is not.

Details are available in a blog post by Martín Escardó.

The Church–Turing thesis

The **Church–Turing thesis** states:

> If a function $f : \mathbb{N} \to \mathbb{N}$ is computable in the real world, then it's also computable by a Turing machine.

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any function $f : \mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."

means:

> There is a Turing machine which reads the source of a Turing machine computing a function $f : \mathbb{N} \to \mathbb{N}$ as input and outputs the source of a Turing machine which computes f.

That's trivial, echo the input back to the user.

The Church–Turing thesis

The **Church–Turing thesis** states:

> If a function $f : \mathbb{N} \to \mathbb{N}$ is computable in the real world, then it's also computable by a Turing machine.

$\mathrm{Eff}(\mathrm{TM}) \models$ "Any function $f : \mathbb{N} \to \mathbb{N}$ is computable by a Turing machine."

means:

There is a Turing machine which reads the source of a Turing machine computing a function $f : \mathbb{N} \to \mathbb{N}$ as input and outputs the source of a Turing machine which computes f.

That's trivial, echo the input back to the user.

In $\mathrm{Eff}(\mathrm{STM})$ the statement is false.

The statement “any function $\mathbb{N} \to \mathbb{N}$ is computable by a Turing machine” is the *formal Church–Turing thesis.* It is wrong in the standard topos Set.

The meaning of the formal Church–Turing thesis in Eff(STM) is: “There is a super Turing machine which reads the source of a super Turing machine computing a function $f : \mathbb{N} \to \mathbb{N}$ as input and outputs the source of an *ordinary* Turing machine which computes f.” This is false.

The statement is also false in Eff(λC), the effective topos associated to lambda calculus. This is because of different calling conventions; a higher-order lambda term has no access to the syntactic structure of a passed argument.

Whether the formal Church–Turing thesis holds in Eff(RW), the effective topos associated to machines of the real world, depends on the nature of computation in the real world.

The topos Eff(TM) is a nice context for computer science, since in it uncomputable functions are not present. By contrast, in classical mathematics, there are many functions which are not computable. Two of the most famous ones are the “halting predicate” and the “Busy Beaver function”:

$$H(n) = \begin{cases} 1, & \text{if the } n\text{'th Turing machine halts,} \\ 0, & \text{otherwise.} \end{cases}$$

$$\begin{aligned} BB(n) = {} & \text{maximal number of steps which a halting Turing machine} \\ & \text{with } n \text{ states performs before halting.} \end{aligned}$$

Can’t we use exactly the same definitions in the effective topos, thereby contradicting the theorem that all functions in the effective topos are computable?

The apparent paradox is resolved in the following way. The theorem only states that total functions $\mathbb{N} \to \mathbb{N}$ are computable. However, to verify that H and BB are indeed such total functions, the law of excluded middle is necessary:

For H, in order to be able to make the case distinction.

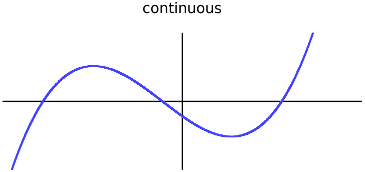
For BB, because implicitly the lemma that every "subfinite" set of natural numbers contains a maximal element was used. This lemma depends on the law of excluded middle.

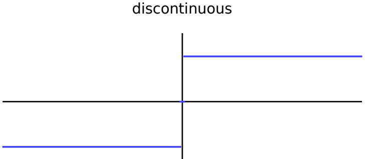
Automatic continuity

The following statement is wildly **false** in Set:

"Every function $f : \mathbb{R} \to \mathbb{R}$ is continuous."

A function f is **continuous** if and only if, for calculating $f(x)$ to finitely many digits, finitely many digits of x suffice.



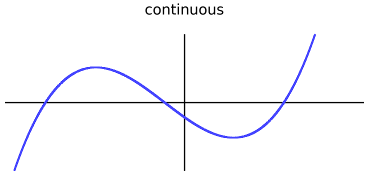


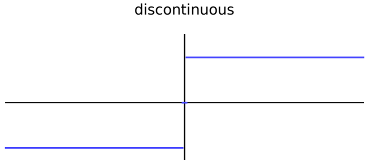
Automatic continuity

The following statement is wildly **false** in Set:

"Every function $f : \mathbb{R} \to \mathbb{R}$ is continuous."

A function f is **continuous** if and only if, for calculating $f(x)$ to finitely many digits, finitely many digits of x suffice.



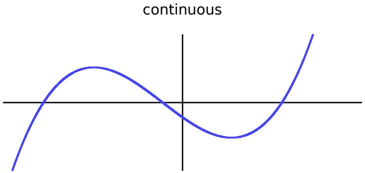


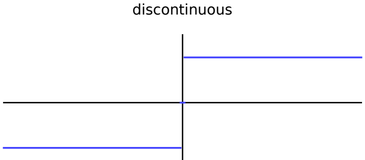
Automatic continuity

The following statement is wildly **false** in Set:

"Every function $f : \mathbb{R} \to \mathbb{R}$ is continuous."

A function f is **continuous** if and only if, for calculating $f(x)$ to finitely many digits, finitely many digits of x suffice.





Don't there obviously exist discontinuous functions? Like the sign function?

$$\operatorname{sgn} : \mathbb{R} \to \mathbb{R}, \quad x \mapsto \begin{cases} -1, & \text{if } x < 0, \\ 0, & \text{if } x = 0, \\ 1, & \text{if } x > 0. \end{cases}$$

Without the presence of the law of excluded middle, the situation is not so trivial: Without LEM, one cannot show that this rule defines a total function $\mathbb{R} \to \mathbb{R}$, because without LEM, the lemma "every real number is either < 0, $= 0$, or > 0" can't be shown. (This lemma implies the weaker statement that any real number is either zero or not zero. We've already seen that this statement is false in Eff(TM).)

Without LEM, the rule only defines a function $M \to \mathbb{R}$, where $M = \{x \in \mathbb{R} \mid x < 0 \vee x = 0 \vee x > 0\}$.

The real numbers can be constructed in several different ways, for instance using Cauchy sequences or Dedekind cuts. Because the axiom of dependent choice holds in Eff(TM), Eff(STM), and Eff(RW), all those constructions coincide.

Externally, a real number in any of these toposes is given by a machine which produces coherent, arbitrarily good approximations. For instance, one could ask such a machine for approximations which are correct to three, seven, or ten digits and obtain the answers

$$3.1417777777, \quad 3.1415926777, \quad \text{and} \quad 3.1415926535.$$

Machines which compute different, but equally good approximations represent the same real number.

The fine print: By “correct to n digits” we mean “distance at most 10^{-n} to the true value”. We don’t literally refer to the first n digits after the decimal point (which aren’t well-defined anyway, $0.999\ldots = 1$).

For instance, the approximation 0.999 of the number 1 is correct to three digits while 0.998 is only correct to two digits.

The meaning of the statement "all functions $\mathbb{R} \to \mathbb{R}$ are continuous" in the effective topos is: There is a machine M which takes

1. a machine A which computes a function $f : \mathbb{R} \to \mathbb{R}$,
2. a machine X which represents a real number x, and
3. a natural number n

as inputs and outputs a natural number m with the following property: For any real number $\tilde{x}$ which is the same as x to m digits, the numbers $f(x)$ and $f(\tilde{x})$ are the same to n digits.

In the case of real world machines, one can proceed as follows to construct such a machine M. Given A, X, and n, the machine M should call A with a slight variant X' of X: The machine X' should exhibit the same input/output behaviour as X, but on each call X' should tell M using a private communication channel how many digits were asked for. Since X' has the same input/output behaviour as X, the machine A is bound by contract to react to X' in the same way as it reacts to X.

If the real world is such that only finitely many computational steps can be performed in finite time, then M can determine a suitable number m as follows: It just has to wait till A has computed $f(x)$ to n digits. It can then look at its log to see how many digits of x A used for the computation. For any number $\tilde{x}$ which is equal to x to this amount of digits, the number $f(\tilde{x})$ produced by A is the same as $f(x)$ to n digits.

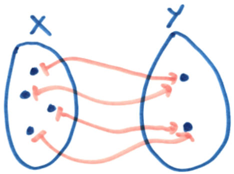
Not a fan of real numbers? The same phenomenon is visible at other types, for instance for functions $f : \mathbb{B}^{\mathbb{N}} \to \mathbb{B}$, where $\mathbb{B} = \{0, 1\}$ as before. Such a function is continuous if and only if, for any $x \in \mathbb{B}^{\mathbb{N}}$ (that is, any function $x : \mathbb{N} \to \mathbb{B}$) there exists a number m such that $f(x)$ depends only on the first m values of x.

In $\mathrm{Eff}(\mathrm{TM})$, the statement "every function $f : \mathbb{B}^{\mathbb{N}} \to \mathbb{B}$ is continuous" is true.

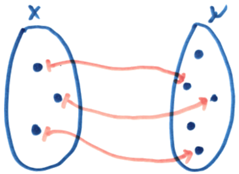
Curious size phenomena

There is no surjection $\mathbb{N} \to \mathbb{N}^{\mathbb{N}}$; the set $\mathbb{N}^{\mathbb{N}}$ of functions $\mathbb{N} \to \mathbb{N}$ is much larger than $\mathbb{N}$.

A corollary in classical logic is: There is no injection $\mathbb{N}^{\mathbb{N}} \to \mathbb{N}$. This expresses the same intuition about the relative sizes.



a surjection
"X is greater than Y"

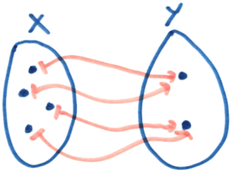


an injection
"X is smaller than Y"

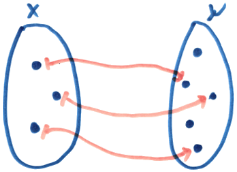
Curious size phenomena

There is no surjection $\mathbb{N} \to \mathbb{N}^{\mathbb{N}}$; the set $\mathbb{N}^{\mathbb{N}}$ of functions $\mathbb{N} \to \mathbb{N}$ is much larger than $\mathbb{N}$.

A corollary in classical logic is: There is no injection $\mathbb{N}^{\mathbb{N}} \to \mathbb{N}$. This expresses the same intuition about the relative sizes.



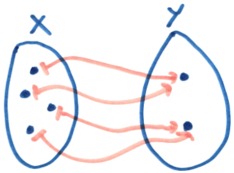
a surjection
"X is greater than Y"



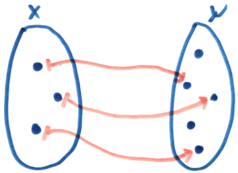
an injection
"X is smaller than Y"

In Eff(STM), there **is such an injection**.

Curious size phenomena



a surjection
“X is greater than Y”



an injection
“X is smaller than Y”

$\mathrm{Eff}(\mathrm{STM}) \models$ “There exists an injection $\mathbb{N}^{\mathbb{N}} \to \mathbb{N}$.”
means:

There is a super Turing machine which inputs the source of a super Turing machine A computing a function $\mathbb{N} \to \mathbb{N}$ and outputs a number $n(A)$ such that $n(A) = n(B)$ if and only if A and B compute the same function.

Curious size phenomena

$\mathrm{Eff}(\mathrm{STM}) \models$ "There exists an injection $\mathbb{N}^{\mathbb{N}} \to \mathbb{N}$."

means:

> There is a super Turing machine which inputs the source of a super Turing machine A computing a function $\mathbb{N} \to \mathbb{N}$ and outputs a number $n(A)$ such that $n(A) = n(B)$ if and only if A and B compute the same function.

This statement is witnessed by following super Turing machine:

Read the source of a super Turing machine A from the tape. Simulate all super Turing machines in a dovetailing fashion. As soon a machine is found which has the same input/output behaviour as A, output the number of this machine and halt.

The number computed by that super Turing machine depends on the input/output behaviour of A, the chosen order of all super Turing machines, and on details on the way the interleaving simulation works – but it does *not* depend on the implementation of A.

The search terminates since there is at least one super Turing machine which halts on any natural number input and shows the same input/output behaviour as A: A itself.

This injection was found by Andrej Bauer.

Just because it is so nice, here is a constructively valid proof that there is no surjection $\mathbb{N} \to \mathbb{N}^{\mathbb{N}}$:

Let $s : \mathbb{N} \to \mathbb{N}^{\mathbb{N}}$ be an arbitrary map. We want to verify that s is not surjective. For this, we consider the map $f : \mathbb{N} \to \mathbb{N}$ given by

$$f(n) := s(n)(n) + 1.$$

This element $f \in \mathbb{N}^{\mathbb{N}}$ is not in the image of s: If there existed a number m such that $s(m) = f$, then

$$s(m)(m) = f(m) = s(m)(m) + 1.$$

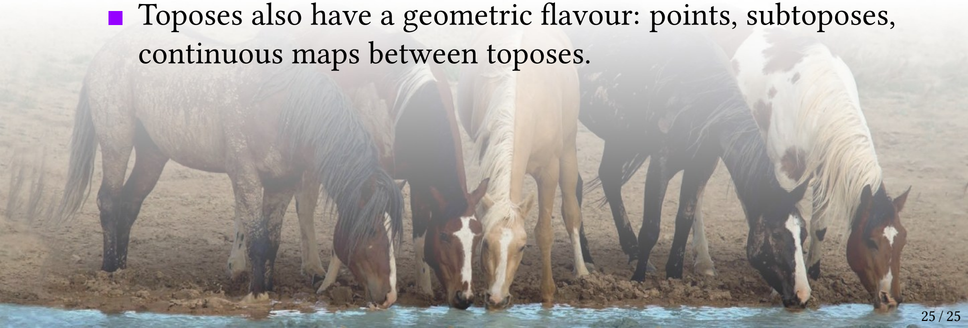
When thinking about size issues, it's important to not mix up the external and internal point of view – else one encounters *Skolem's paradox*: How is it possible that $\mathrm{Eff}(\mathrm{TM})$ doesn't contain a surjection $\mathbb{N} \to \mathbb{N}^{\mathbb{N}}$, even though the object $\mathbb{N}^{\mathbb{N}}$ of the effective topos contains only computable functions and there are only countably many of those?

It's true that the underlying set of the object $\mathbb{N}^{\mathbb{N}}$ of $\mathrm{Eff}(\mathrm{TM})$ is countable, so that there is a surjection (even bijection) $\mathbb{N} \to \mathbb{N}^{\mathbb{N}}$.

But any such surjection is either not computable, therefore not contained in $\mathrm{Eff}(\mathrm{TM})$, or is computable, but in a way that there is no computable witness of its surjectivity. Therefore $\mathbb{N}^{\mathbb{N}}$ looks like an uncountable set from the internal point of view $\mathrm{Eff}(\mathrm{TM})$, just as Cantor's theorem predicts.

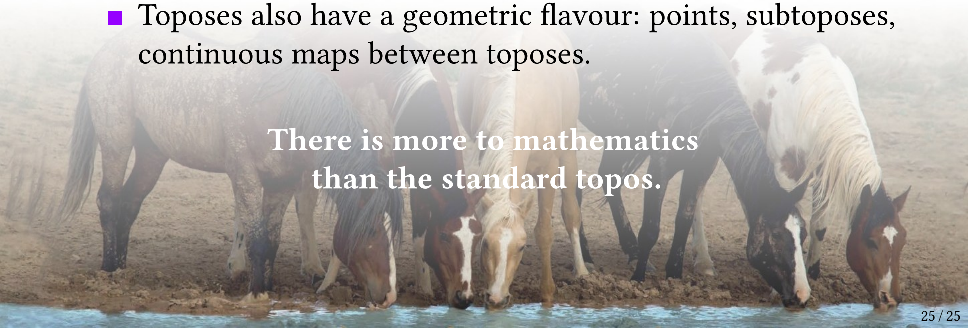
Wrapping up

- Effective toposes are a good vehicle for studying the nature of computation.
- Effective toposes build links between constructive mathematics and programming.
- Toposes allow for curious dream axioms.
- Toposes also have a geometric flavour: points, subtoposes, continuous maps between toposes.



Wrapping up

- Effective toposes are a good vehicle for studying the nature of computation.
- Effective toposes build links between constructive mathematics and programming.
- Toposes allow for curious dream axioms.
- Toposes also have a geometric flavour: points, subtoposes, continuous maps between toposes.



There is more to mathematics than the standard topos.

Any topos has a *largest dense subtopos.* The largest dense subtopos of Eff(TM) is Set, the standard topos. It is related to the *double negation modality.*

For instance, the statement "for any number *n* there is a prime $p > n$" holds in Eff(TM) for the nontrivial reason that there exists a Turing machine which can find arbitrarily large prime numbers. The weaker statement "for any number *n* there is *not not* a prime $p > n$" holds in Eff(TM) as well, but doesn't require a witness by a machine.

If one prefixes all occurences of "$\exists$" and "$\vee$" in a formula φ with "$\neg\neg$"'s, then it loses its computational content. This modified formula holds in Eff(TM) if and only if the original formula holds in Set.

For any *oracle* L (as studied in computer science), the effective topos associated to Turing machines with access to L is a subtopos of $\mathrm{Eff}(\mathrm{TM})$. It too corresponds to a certain modal operator.

In the *smooth topos*, employed in synthetic differential geometry, the following statement holds:

Axiom of microaffinity: For any function $f : \Delta \to \mathbb{R}$, where $\Delta = \{\varepsilon \in \mathbb{R} \mid \varepsilon^2 = 0\}$, there exists a unique pair (a, b) of real numbers such that

$$f(\varepsilon) = a + b\varepsilon$$

for all $\varepsilon \in \Delta$.