Stream representations of real numbers

Herman Geuvers

Radboud University Nijmegen/ Technical University Eindhoven

November 1, 2010

Overview

- Real numbers
- Representaions of real numbers via streams
- Stream computability: TTE
- Computability of $f : \mathbb{R} \rightarrow \mathbb{R}$
- Concrete representations and algorithms

Overview

- Real numbers
- Representaions of real numbers via streams
- Stream computability: TTE
- Computability of $f : \mathbb{R} \to \mathbb{R}$
- Concrete representations and algorithms

Based on (a.o.)
Weihrauch - Computable Analysis (An Introduction), Springer EATCS series 2000.
Niqui - Formalising Exact Real Arithmetic (Representations, Algorithms, Proofs), PhD thesis, Radboud University Nijmegen, 2004.

Related work / what I will not talk about

Detailed algorithms for exact real arithmetic

- Edalat, Potts et al.

Related work / what I will not talk about

Detailed algorithms for exact real arithmetic

- Edalat, Potts et al.

Formalization in Coq

- Niqui, Bertot, Julien

Related work / what I will not talk about

Detailed algorithms for exact real arithmetic

- Edalat, Potts et al.

Formalization in Coq

- Niqui, Bertot, Julien

Other approaches

- O'Connor: use $f : \mathbb{Q}^+ \to \mathbb{Q}$ in stead of $f : \mathbb{N} \to \mathbb{Q}$
- Pasca: formalize interval arithmetic
- Work with reals axiomatically, e.g. Mayero (Coq standard lib, classical); Cruz-Filipe (CoRN, constructive analysis)

Real numbers

- uncountably many
- only countably many are representable via syntax

Real numbers

- uncountably many
- only countably many are representable via syntax
- $\mathbb{Q}$ is countable and dense in $\mathbb{R}$.
- So: use a countable set of approximations to elements in $\mathbb{R}$.

Defining real numbers via rational approximations

For example:

$$
\begin{aligned}
e &:= \sum_{i=0}^{\infty} \frac{1}{i!} = \lim_{n\to\infty} \sum_{i=0}^{n} \frac{1}{i!} \\
e &:= \lim_{n\to\infty} (1 + \frac{1}{n})^n \\
\frac{\pi^2}{6} &:= \sum_{i=0}^{\infty} \frac{1}{i^2} = \lim_{n\to\infty} \sum_{i=0}^{n} \frac{1}{i^2}
\end{aligned}
$$

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (I): Cauchy sequences

Sequence of rational numbers $q_0, q_1, q_2, \ldots$ that converges.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N(|q_m - q_N| < \frac{1}{k})$$
$$\forall \epsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N(|q_m - q_N| < \epsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m, p(|q_{N+m} - q_{N+p}| < \frac{1}{k})$$

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (I): Cauchy sequences

Sequence of rational numbers $q_0, q_1, q_2, \ldots$ that converges.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \tfrac{1}{k})$$
$$\forall \epsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \epsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m, p (|q_{N+m} - q_{N+p}| < \tfrac{1}{k})$$

Option: Express N in terms of k via a function $\beta(k)$

$$\forall k \in \mathbb{N}^+ \forall m, p (|q_{\beta(k)+m} - q_{\beta(k)+p}| < \frac{1}{k})$$

Fundamental sequence: pair of $(q_i)_{i \in \mathbb{N}}$ and $\beta : \mathbb{N} \to \mathbb{N}$.

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (I): Cauchy sequences

Sequence of rational numbers $q_0, q_1, q_2, \ldots$ that converges.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \tfrac{1}{k})$$
$$\forall \epsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - q_N| < \epsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m, p (|q_{N+m} - q_{N+p}| < \tfrac{1}{k})$$

Option: Express N in terms of k via a function $\beta(k)$

$$\forall k \in \mathbb{N}^+ \forall m, p (|q_{\beta(k)+m} - q_{\beta(k)+p}| < \frac{1}{k})$$

Fundamental sequence: pair of $(q_i)_{i \in \mathbb{N}}$ and $\beta : \mathbb{N} \to \mathbb{N}$.
Decimal notation

$$3,\ 3.1,\ 3.14,\ 3.141,\ 3.1415,\ 3.14159,\ \ldots$$

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (II): Decreasing intervals

Sequence of rational intervals $[p_0, q_0] \supseteq [p_1, q_1] \supseteq [p_2, q_2], \ldots$ that converges.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N(|q_m - p_m| < \frac{1}{k})$$
$$\forall \epsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N(|q_m - p_m| < \epsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m(|q_{N+m} - p_{N+m}| < \frac{1}{k})$$

Sequence of rational intervals $[p_0, q_0] \supseteq [p_1, q_1] \supseteq [p_2, q_2], \ldots$ that converges.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - p_m| < \frac{1}{k})$$
$$\forall \epsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N (|q_m - p_m| < \epsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m (|q_{N+m} - p_{N+m}| < \frac{1}{k})$$

Option: Express N in terms of k via a function $\beta(k)$

$$\forall k \in \mathbb{N}^+ \forall m (|q_{\beta(k)+m} - p_{\beta(k)+m}| < \frac{1}{k})$$

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (II): Decreasing intervals

Sequence of rational intervals $[p_0, q_0] \supseteq [p_1, q_1] \supseteq [p_2, q_2], \ldots$ that converges.

$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m \geq N(|q_m - p_m| < \frac{1}{k})$$
$$\forall \epsilon \in \mathbb{Q}^+ \exists N \in \mathbb{N} \forall m \geq N(|q_m - p_m| < \epsilon)$$
$$\forall k \in \mathbb{N}^+ \exists N \in \mathbb{N} \forall m(|q_{N+m} - p_{N+m}| < \frac{1}{k})$$

Option: Express N in terms of k via a function $\beta(k)$

$$\forall k \in \mathbb{N}^+ \forall m(|q_{\beta(k)+m} - p_{\beta(k)+m}| < \frac{1}{k})$$

Decimal notation can be seen as an interval representation

3	3.1	3.14	3.141	3.1415
$[3, 4]$	$[3.1, 3.2]$	$[3.14, 3.15]$	$[3.141, 3.142]$	$[3.1415, 3.1416]$

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (III): Dedekind Cuts

A partition (A, B) of $\mathbb{Q}$ such that

- $\forall x \in A\, \forall y \in B (x < y)$,
- $\forall x \in A\, \exists z \in A (x < z)$,
- $A \neq \emptyset$, $B \neq \emptyset$

(Or variations)

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (III): Dedekind Cuts

A partition (A, B) of $\mathbb{Q}$ such that

- $\forall x \in A\, \forall y \in B (x < y)$,
- $\forall x \in A\, \exists z \in A (x < z)$,
- $A \neq \emptyset$, $B \neq \emptyset$

(Or variations)
So, a real number can be defined via a (unique) set. E.g.

$$A := \{x \in \mathbb{Q} | x^2 < 2 \vee x \leq 0\}$$

Defining $\mathbb{R}$ out of $\mathbb{Q}$ (III): Dedekind Cuts

A partition (A, B) of $\mathbb{Q}$ such that

- $\forall x \in A\, \forall y \in B (x < y)$,
- $\forall x \in A\, \exists z \in A (x < z)$,
- $A \neq \emptyset$, $B \neq \emptyset$

(Or variations)
So, a real number can be defined via a (unique) set. E.g.

$$A := \{x \in \mathbb{Q} | x^2 < 2 \vee x \leq 0\}$$

In concrete situations, a real number is defined via a property.
We need logic to prove equality of real numbers.

Equality on real numbers

Real number equality $\neq$ Stream equality

- $([-\frac{1}{2^k}, \frac{1}{2^k}])_{k\in\mathbb{N}} = ([-\frac{1}{3^k}, \frac{1}{3^k}])_{k\in\mathbb{N}}$
- $0.999\ldots = 1.000\ldots$

There is no hope in deciding that two real number representations are equal.

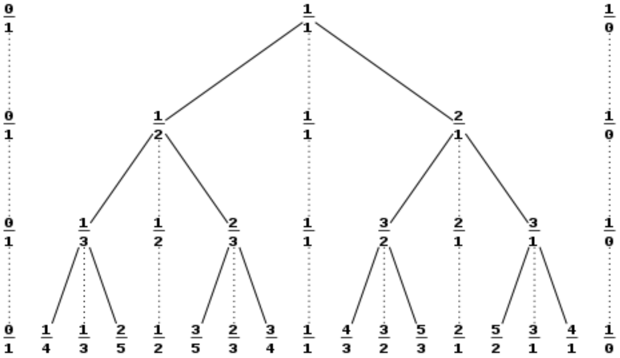
Equality on real numbers

Real number equality $\neq$ Stream equality

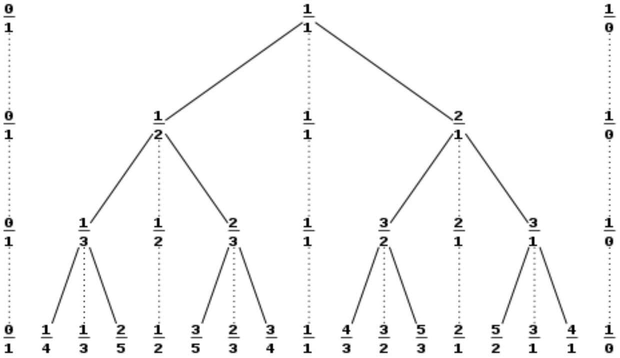
- $([-\frac{1}{2^k}, \frac{1}{2^k}])_{k\in\mathbb{N}} = ([-\frac{1}{3^k}, \frac{1}{3^k}])_{k\in\mathbb{N}}$
- $0.999\ldots = 1.000\ldots$

There is no hope in deciding that two real number representations are equal.
Would it help to remove redundancy in representations?

Stern-Brocot tree: unique representation of rational numbers

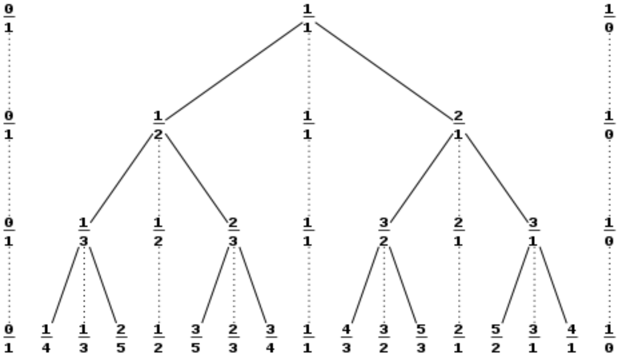


Stern-Brocot tree: unique representation of rational numbers



- Compute new element from “ancestors” $\frac{p}{q}$ and $\frac{n}{m}$ by taking the mediant: $\frac{p+n}{q+m}$

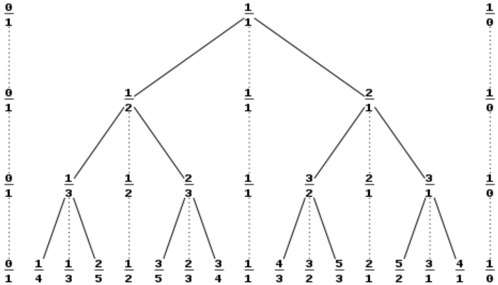
Stern-Brocot tree: unique representation of rational numbers



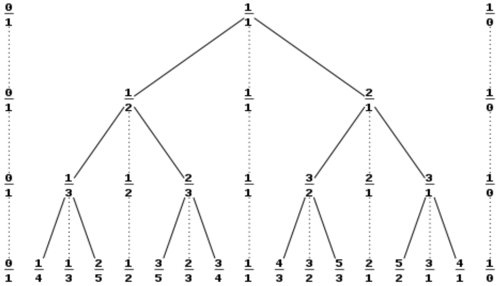
The Stern-Brocot tree gives a unique representation of $\mathbb{Q}^+$:

- For every $\frac{p}{q}$ in the tree, p and q are coprime.
- Every $x \in \mathbb{Q}^+$ occurs exactly once in the tree.

Stern-Brocot tree for real numbers

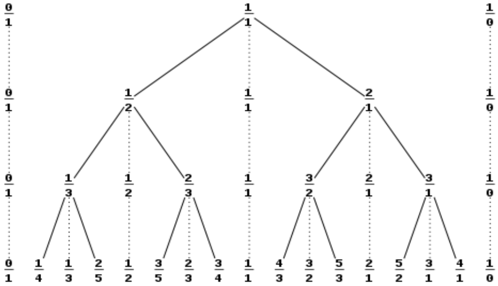


Stern-Brocot tree for real numbers



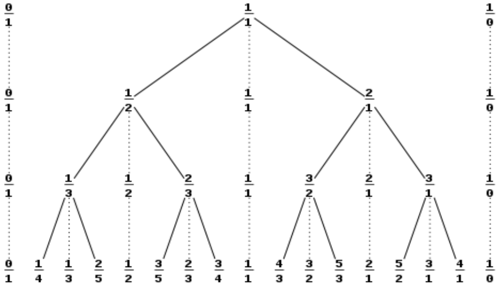
- A path in the tree gives a rational number via $[\![\langle - \rangle]\!] = 1$, $[\![L\sigma]\!] = \frac{[\![\sigma]\!]}{[\![\sigma]\!]+1}$, $[\![R\sigma]\!] = [\![\sigma]\!] + 1$.

Stern-Brocot tree for real numbers



- A path in the tree gives a rational number via $[\![\langle - \rangle]\!] = 1$, $[\![L\sigma]\!] = \frac{[\![\sigma]\!]}{[\![\sigma]\!]+1}$, $[\![R\sigma]\!] = [\![\sigma]\!] + 1$.
- Two "base functions" $L(x) = \frac{x}{x+1}$ and $R(x) = x + 1$,

Stern-Brocot tree for real numbers



- A path in the tree gives a rational number via $[\![\langle - \rangle]\!] = 1$, $[\![L\sigma]\!] = \frac{[\![\sigma]\!]}{[\![\sigma]\!]+1}$, $[\![R\sigma]\!] = [\![\sigma]\!] + 1$.
- Two "base functions" $L(x) = \frac{x}{x+1}$ and $R(x) = x + 1$,
- An infinite path in the tree $d_0, d_1, d_2, \ldots$ represents $\bigcap_{i\in\mathbb{N}} d_0(d_1 \ldots d_i([0,\infty]) \ldots)$

Redundancy in representation

Are representations with little redundancy better?
E.g. Stern Brocot tree, continued fractions (with certain side conditions) have little redundancy.

Redundancy in representation

Are representations with little redundancy better?
E.g. Stern Brocot tree, continued fractions (with certain side conditions) have little redundancy.
No!

- Too little redundancy makes certain functions non-computable.
- We want every $x \in \mathbb{R}$ to have countably many representations.

What is stream computability?

Type-2 Turing Machines and TTE, Type-2 Theory of Effectivity [Weihrauch, Grzegorczyk]

What is stream computability?

Type-2 Turing Machines and TTE, Type-2 Theory of Effectivity [Weihrauch, Grzegorczyk]

- Normal multi-tape Turing machine over finite input alphabet Σ
- Special input tape with infinite stream of symbols as input,
- Special output tape where symbols from Σ can only be written. (Never erase.)
  Here the output $\in \Sigma^\omega$ is generated.

TTE computability

- **Def** Turing Machine M computes $s \in \Sigma^\omega$ if M with empty input tape runs forever and produces s on the output tape.

TTE computability

- **Def** Turing Machine M computes $s \in \Sigma^\omega$ if
  M with empty input tape runs forever and produces s on the output tape.
- **Def** Turing Machine M computes $f : \Sigma^\omega \to \Sigma^\omega$ if for all $s \in \Sigma^\omega$:
  M with s on the input tape runs forever and produces $f(s)$ on the output tape.

TTE computability

- **Def** Turing Machine M computes $s \in \Sigma^\omega$ if
  M with empty input tape runs forever and produces s on the output tape.
- **Def** Turing Machine M computes $f : \Sigma^\omega \to \Sigma^\omega$ if for all $s \in \Sigma^\omega$:
  M with s on the input tape runs forever and produces $f(s)$ on the output tape.
- NB. If M on s only produces a finite amount of output, it is supposed to be undefined.
  So the definition generalizes to partial $f : \Sigma^\omega \to \Sigma^\omega$.

TTE computability

- **Def** Turing Machine M computes $s \in \Sigma^\omega$ if
  M with empty input tape runs forever and produces s on the output tape.
- **Def** Turing Machine M computes $f : \Sigma^\omega \to \Sigma^\omega$ if for all $s \in \Sigma^\omega$:
  M with s on the input tape runs forever and produces $f(s)$ on the output tape.
- NB. If M on s only produces a finite amount of output, it is supposed to be undefined.
  So the definition generalizes to partial $f : \Sigma^\omega \to \Sigma^\omega$.
- **Def** $f : \Sigma^\omega \to \Sigma^\omega$ is computable if there is a Turing machine M that computes it, etc.

Σ^ω as a metric space

- Σ^ω has the Cantor topology: basic opens $p\Sigma^\omega$ with $p \in \Sigma^*$

Σ^ω as a metric space

- Σ^ω has the Cantor topology: basic opens $p\Sigma^\omega$ with $p \in \Sigma^*$
- Metric on Σ^ω:

$$
\begin{aligned}
d(s,t) &:= 2^{-n} \text{ if } n \text{ is the smallest for which } s(n) \neq t(n) \\
d(s,s) &:= 0 \text{ otherwise}
\end{aligned}
$$

Σ^ω as a metric space

- Σ^ω has the Cantor topology: basic opens $p\Sigma^\omega$ with $p \in \Sigma^*$
- Metric on Σ^ω:

$$\begin{array}{rcl} d(s,t) & := & 2^{-n} \text{ if } n \text{ is the smallest for which } s(n) \neq t(n) \\ d(s,s) & := & 0 \text{ otherwise} \end{array}$$

- In general, dom(f) is the infinite intersection of open sets. E.g. $f(0:s) = f(s)$, $f(1:s) = 1:f(s)$, $\text{dom}(f) = \{s | s \text{ contains } \infty \text{ many } 1\}$.

Σ^ω as a metric space

- Σ^ω has the Cantor topology: basic opens $p\Sigma^\omega$ with $p \in \Sigma^*$
- Metric on Σ^ω:

$$\begin{aligned} d(s,t) &:= 2^{-n} \text{ if } n \text{ is the smallest for which } s(n) \neq t(n) \\ d(s,s) &:= 0 \text{ otherwise} \end{aligned}$$

- In general, dom(f) is the infinite intersection of open sets.
E.g. $f(0 : s) = f(s)$, $f(1 : s) = 1 : f(s)$,
dom$(f) = \{s | s \text{ contains } \infty \text{ many } 1\}$.
Define $A_i := \bigcup\{p\Sigma^\omega | p \text{ contains } i \text{ times a } 1\}$.
$B := \bigcap_i A_i$ is the domain of f.

Σ^ω as a metric space

- Σ^ω has the Cantor topology: basic opens $p\Sigma^\omega$ with $p \in \Sigma^*$
- Metric on Σ^ω:

$$\begin{aligned} d(s,t) &:= 2^{-n} \text{ if } n \text{ is the smallest for which } s(n) \neq t(n) \\ d(s,s) &:= 0 \text{ otherwise} \end{aligned}$$

- In general, $\text{dom}(f)$ is the infinite intersection of open sets.
  E.g. $f(0:s) = f(s)$, $f(1:s) = 1 : f(s)$,
  $\text{dom}(f) = \{s | s \text{ contains } \infty \text{ many } 1\}$.
  Define $A_i := \bigcup\{p\Sigma^\omega | p \text{ contains } i \text{ times a } 1\}$.
  $B := \bigcap_i A_i$ is the domain of f.
- NB. X is a Π^0_2-set iff $X = \text{dom}(f)$ for some $f : \Sigma^* \to \Sigma^\omega$.

Every computable function is continuous

Theorem Every $f : \Sigma^\omega \to \Sigma$ that is computable, is also continuous.
Proof Let $s \in \Sigma^\omega$,$\epsilon > 0$. To prove:
$\exists\delta\forall t(|s - t| < \delta \to |f(s) - f(t)| < \epsilon)$.
Take p with $2^{-p} < \epsilon$ and compute $f(s)$ until p outputs have been written on the output tape.

Every computable function is continuous

Theorem Every $f : \Sigma^\omega \to \Sigma$ that is computable, is also continuous.

Proof Let $s \in \Sigma^\omega$,$\epsilon > 0$. To prove:
$\exists\delta\forall t(|s - t| < \delta \to |f(s) - f(t)| < \epsilon)$.
Take p with $2^{-p} < \epsilon$ and compute $f(s)$ until p outputs have been written on the output tape.
We have now read a finite number of inputs, say the first q.
Take $\delta := 2^{-q}$

Every computable function is continuous

Theorem Every $f : \Sigma^\omega \to \Sigma$ that is computable, is also continuous.
Proof Let $s \in \Sigma^\omega$,$\epsilon > 0$. To prove:
$\exists\delta\forall t(|s - t| < \delta \to |f(s) - f(t)| < \epsilon)$.
Take p with $2^{-p} < \epsilon$ and compute $f(s)$ until p outputs have been written on the output tape.
We have now read a finite number of inputs, say the first q.
Take $\delta := 2^{-q}$
Now for all t, if $|s - t| < \delta$, then t begins with the same q inputs as s, so $|f(s) - f(t)| < 2^{-p} < \epsilon$.

Domain theory: a slightly more general view

Domain D of finite and infinite sequences over Σ.
($D := \Sigma^* \cup \Sigma^\omega$.)

$$s \sqsubseteq t \text{ iff } \forall i < \text{length}(s)(s_i = t_i)$$

Domain theory: a slightly more general view

Domain D of finite and infinite sequences over Σ.
($D := \Sigma^* \cup \Sigma^\omega$.)

$$s \sqsubseteq t \text{ iff } \forall i < \text{length}(s)(s_i = t_i)$$

Now every Turing machine M computes a total function from D to D

Domain theory: a slightly more general view

Domain D of finite and infinite sequences over Σ.
($D := \Sigma^* \cup \Sigma^\omega$.)

$$s \sqsubseteq t \text{ iff } \forall i < \text{length}(s)(s_i = t_i)$$

Now every Turing machine M computes a total function from D to D

Theorem If $f : D \to D$ is computable, it is

- monotonic: if $s \sqsubseteq t$, then $f(s) \sqsubseteq f(t)$,
- continuous: $f(\sqcup_i t_i) = \sqcup_i f(t_i)$.

Domain theory: a slightly more general view

Domain D of finite and infinite sequences over Σ.
($D := \Sigma^* \cup \Sigma^\omega$.)

$$s \sqsubseteq t \text{ iff } \forall i < \text{length}(s)(s_i = t_i)$$

Now every Turing machine M computes a total function from D to D

Theorem If $f : D \to D$ is computable, it is

- monotonic: if $s \sqsubseteq t$, then $f(s) \sqsubseteq f(t)$,
- continuous: $f(\sqcup_i t_i) = \sqcup_i f(t_i)$.

(Proof of $f(\sqcup_i t_i) \sqsubseteq \sqcup_i f(t_i)$ is similar to proof of continuity via metric space.)

Computing with infinite streams that represent reals

Computing with infinite streams that represent reals

We take the decimal representation and compute the "double" function:

$$0,1234567\ldots \quad \xrightarrow{\times 2} \quad ??$$

$$\begin{array}{rcl}
0.1 & \xrightarrow{\times 2} & 0. \\
0.12 & \xrightarrow{\times 2} & 0.2 \\
0.123 & \xrightarrow{\times 2} & 0.24 \\
0.1234 & \xrightarrow{\times 2} & 0.246 \\
0.12345 & \xrightarrow{\times 2} & 0.2469 \\
0.123456 & \xrightarrow{\times 2} & 0.24691 \\
 & \ldots &
\end{array}$$

"Look ahead" $L_f(y)(k) = k + 1$

Problem with computing with decimal representation

We compute the “triple function:

$$0,3333\ldots \qquad \stackrel{\times 3}{\longrightarrow} \qquad ??$$

$$\begin{array}{rcl} 0.3 & \stackrel{\times 3}{\longrightarrow} & ? \\ 0.33 & \stackrel{\times 3}{\longrightarrow} & ? \\ 0.333 & \stackrel{\times 3}{\longrightarrow} & ? \\ & \ldots & \end{array}$$

Problem with computing with decimal representation

We compute the "triple function:

$$0,3333\ldots \xrightarrow{\times 3} ??$$

$$\begin{array}{rcl} 0.3 & \xrightarrow{\times 3} & ? \\ 0.33 & \xrightarrow{\times 3} & ? \\ 0.333 & \xrightarrow{\times 3} & ? \\ & \ldots & \end{array}$$

$0.333 \xrightarrow{\times 3}$ 0.9 wrong in case 0.3334

$0.333 \xrightarrow{\times 3}$ 1. wrong in case 0.3332

The decimal representation is not good

With the decimal representation, we can't compute addition, multiplication etcetera.
So, representing reals via their infinite decimal representation is not good.

The decimal representation is not good

With the decimal representation, we can't compute addition, multiplication etcetera.
So, representing reals via their infinite decimal representation is not good.
Brouwer knew this already:
"Besitzt jede reelle Zahl eine Dezimalbruchentwikklung?"
(Mathematische Annalen, 1921)

A good representation has redundancy

Decimal representation:

$$\ldots \mid \quad 0.0 \quad \mid \quad 0.1 \quad \mid \quad 0.2 \quad \mid \ldots$$

0.0 denotes [0.0, 0.1] and 0.1 denotes [0.1, 0.2].
What if the number we are trying to approximate turns out to be 0.1?

A good representation has redundancy

Decimal representation:

$$\ldots \mid \quad 0.0 \quad \mid \quad 0.1 \quad \mid \quad 0.2 \quad \mid \ldots$$

0.0 denotes [0.0, 0.1] and 0.1 denotes [0.1, 0.2].
What if the number we are trying to approximate turns out to be 0.1?
Solution:: Add one extra digit: -1

...	[0.0]		[0.2]		...
...		[0.1]		[0.3]	...

A good representation has redundancy

Decimal representation:

...	0.0	0.1	0.2	...

0.0 denotes [0.0, 0.1] and 0.1 denotes [0.1, 0.2].
What if the number we are trying to approximate turns out to be 0.1?
Solution:: Add one extra digit: -1

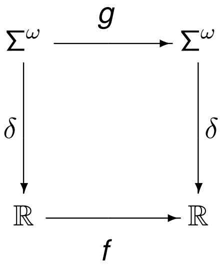
...	[0.0]		[0.2]		...
...		[0.1]		[0.3]	...

$$[\![d_0 d_1 d_2 d_3 \ldots]\!] := \sum_{i=0}^{\infty} d_i \times 10^{-i}$$

Representation systems

Definition A representation is a surjective $\delta : \Sigma^\omega \to \mathbb{R}$ (possibly partial).
Definition $x \in \mathbb{R}$ is δ-computable if $x = \delta(s)$ for some $s \in \Sigma^\omega$.

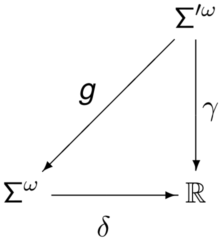
Definition A representation is a surjective $\delta : \Sigma^\omega \to \mathbb{R}$ (possibly partial).
Definition $x \in \mathbb{R}$ is δ-computable if $x = \delta(s)$ for some $s \in \Sigma^\omega$.
Definition $f : \mathbb{R} \to \mathbb{R}$ is δ-computable if there is a computable $g : \Sigma^\omega \to \Sigma^\omega$ such that $f(\delta(s)) = \delta(g(s))$ for all $s \in \Sigma^\omega$.



Good representation systems

Definition A representation system $\delta : \Sigma^\omega \to \mathbb{R}$ is admissible if

- δ is continuous
- δ is maximal: for every continous representation $\gamma : \Sigma'^\omega \to \mathbb{R}$, there is a continuous $g : \Sigma'^\omega \to \Sigma^\omega$ such that $\gamma = \delta \circ g$.

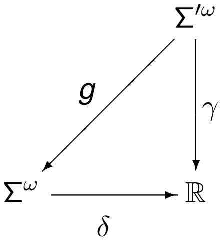


"Every continuous γ can be factored through δ"

Good representation systems

Definition A representation system $\delta : \Sigma^\omega \to \mathbb{R}$ is admissible if

- δ is continuous
- δ is maximal: for every continous representation $\gamma : \Sigma'^\omega \to \mathbb{R}$, there is a continuous $g : \Sigma'^\omega \to \Sigma^\omega$ such that $\gamma = \delta \circ g$.



"Every continuous γ can be factored through δ"

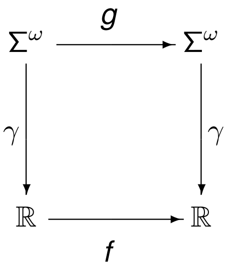
Everything can be restricted to a closed subinterval of $\mathbb{R} \cup \{-\infty, +\infty\}$.

Computable function are continuous

Let $\gamma : \Sigma \to \mathbb{R}$ be an admissible representation.
Theorem If $f : \mathbb{R} \to \mathbb{R}$ is γ-computable, then f is continuous.

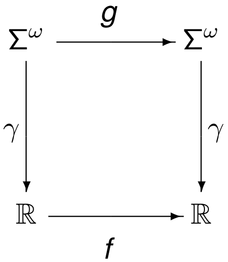
Computable function are continuous

Let $\gamma : \Sigma \to \mathbb{R}$ be an admissible representation.
Theorem If $f : \mathbb{R} \to \mathbb{R}$ is γ-computable, then f is continuous.



Computable function are continuous

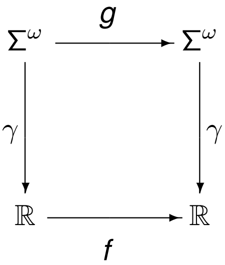
Let $\gamma : \Sigma \to \mathbb{R}$ be an admissible representation.
Theorem If $f : \mathbb{R} \to \mathbb{R}$ is γ-computable, then f is continuous.



Intuition: a computable f produces finite approximations of its output based on finite approximations of its input.

Computable function are continuous

Let $\gamma : \Sigma \to \mathbb{R}$ be an admissible representation.
Theorem If $f : \mathbb{R} \to \mathbb{R}$ is γ-computable, then f is continuous.



Intuition: a computable f produces finite approximations of its output based on finite approximations of its input.
So, inputs that are “close” produce outputs that are “close”.

Comparisons and step functions are not computable

- $c(x,y) := 0$ if $x \leq y$, $c(x,y) := 1$ if $x > y$ is not computable,
- $r(x) := \lfloor x \rfloor$ is not computable.

Comparisons and step functions are not computable

- $c(x,y) := 0$ if $x \leq y$, $c(x,y) := 1$ if $x > y$ is not computable,
- $r(x) := \lfloor x \rfloor$ is not computable.

Of course, we can write an algorithm that, based on a representation of x, tries to compute $\lfloor x \rfloor$, but this algorithm will

1. either occasionally give a wrong answer (e.g. wrongly decide $\lfloor x \rfloor = 1$ based on a finite approximation of x)
2. or not terminate (e.g. wait until it definitely knows that $1 < x < 2$).

Comparisons and step functions are not computable

- $c(x, y) := 0$ if $x \leq y$, $c(x, y) := 1$ if $x > y$ is not computable,
- $r(x) := \lfloor x \rfloor$ is not computable.

Of course, we can write an algorithm that, based on a representation of x, tries to compute $\lfloor x \rfloor$, but this algorithm will

1. either occasionally give a wrong answer (e.g. wrongly decide $\lfloor x \rfloor = 1$ based on a finite approximation of x)
2. or not terminate (e.g. wait until it definitely knows that $1 < x < 2$).

NB. In case (1), different representations of the same x may produce different outputs.

An admissible representation system for $[0, 1]$

For $[0, 1]$, we let $\delta : \{0, 1\}^\omega \to [0, 1]$ be:

$$[\![d_0 d_1 d_2 \ldots]\!] = \sum_{i=0}^{\infty} d_i \times 2^{-i}$$

An admissible representation system for [0, 1]

For [0, 1], we let $\delta : \{0,1\}^\omega \to [0,1]$ be:

$$[\![d_0 d_1 d_2 \ldots]\!] = \sum_{i=0}^{\infty} d_i \times 2^{-i}$$

An alternative way of writing this is

$$\begin{aligned} [\![0 : s]\!] &= \frac{[\![s]\!]}{2} =: \varphi_0([\![s]\!]), \\ [\![1 : s]\!] &= \frac{[\![s]\!] + 1}{2} =: \varphi_1([\![s]\!]). \end{aligned}$$

An admissible representation system for [0, 1]

For [0, 1], we let $\delta : \{0,1\}^\omega \to [0,1]$ be:

$$[\![d_0 d_1 d_2 \ldots]\!] = \sum_{i=0}^{\infty} d_i \times 2^{-i}$$

An alternative way of writing this is

$$\begin{aligned}
[\![0 : s]\!] &= \frac{[\![s]\!]}{2} =: \varphi_0([\![s]\!]), \\
[\![1 : s]\!] &= \frac{[\![s]\!] + 1}{2} =: \varphi_1([\![s]\!]). \\
[\![d_0 d_1 d_2 \ldots]\!] &= \bigcap_{i=0}^{\infty} \varphi_{d_0}(\ldots \varphi_{d_i}([0,1]))
\end{aligned}$$

An admissible representation system for [0, 1]

For [0, 1], we let $\delta : \{0,1\}^\omega \to [0,1]$ be:

$$[\![d_0 d_1 d_2 \ldots]\!] = \sum_{i=0}^{\infty} d_i \times 2^{-i}$$

An alternative way of writing this is

$$\begin{aligned}
[\![0 : s]\!] &= \frac{[\![s]\!]}{2} =: \varphi_0([\![s]\!]), \\
[\![1 : s]\!] &= \frac{[\![s]\!] + 1}{2} =: \varphi_1([\![s]\!]). \\
[\![d_0 d_1 d_2 \ldots]\!] &= \bigcap_{i=0}^{\infty} \varphi_{d_0}(\ldots \varphi_{d_i}([0,1]))
\end{aligned}$$

Now, δ is not admissible. If we add the digit ”2” with $\varphi_2(x) := \frac{x+1/2}{2}$, we have a continous representation, $\gamma : \{0,1,2\}^\omega \to \mathbb{R}$, which does not factor through δ:

An admissible representation system for [0, 1]

For [0, 1], we let $\delta : \{0,1\}^\omega \to [0,1]$ be:

$$[\![d_0 d_1 d_2 \ldots]\!] = \sum_{i=0}^{\infty} d_i \times 2^{-i}$$

An alternative way of writing this is

$$\begin{aligned}
[\![0 : s]\!] &= \frac{[\![s]\!]}{2} =: \varphi_0([\![s]\!]), \\
[\![1 : s]\!] &= \frac{[\![s]\!] + 1}{2} =: \varphi_1([\![s]\!]). \\
[\![d_0 d_1 d_2 \ldots]\!] &= \bigcap_{i=0}^{\infty} \varphi_{d_0}(\ldots \varphi_{d_i}([0,1]))
\end{aligned}$$

Now, δ is not admissible. If we add the digit ”2” with $\varphi_2(x) := \frac{x+1/2}{2}$, we have a continous representation, $\gamma : \{0,1,2\}^\omega \to \mathbb{R}$, which does not factor through δ: $[\![2222\ldots]\!] = \frac{1}{2}$, but there is no computable f such that $f(2222\ldots) = (01111\ldots)$ or $f(2222\ldots) = (100000\ldots)$.

Admissible digit sets

We observe a general procedure for defining a representation for an interval I.

- Define functions $\varphi_1, \ldots, \varphi_k : I \to I$,
- Use them as “digits” for representing the elements of I by

$$[\![d_0 d_1 \ldots]\!] := \bigcap_{i=0}^{\infty} \varphi_{d_0}(\ldots \varphi_{d_i}(I) \ldots)$$

Admissible digit sets

We observe a general procedure for defining a representation for an interval I.

- Define functions $\varphi_1, \ldots, \varphi_k : I \to I$,
- Use them as “digits” for representing the elements of I by

$$[\![d_0 d_1 \ldots]\!] := \bigcap_{i=0}^{\infty} \varphi_{d_0}(\ldots \varphi_{d_i}(I) \ldots)$$

Under which conditions does this method work?

Admissible digit sets

We observe a general procedure for defining a representation for an interval I.

- Define functions $\varphi_1, \dots, \varphi_k : I \to I$,
- Use them as "digits" for representing the elements of I by

$$[\![d_0 d_1 \dots]\!] := \bigcap_{i=0}^{\infty} \varphi_{d_0}(\dots \varphi_{d_i}(I) \dots)$$

Under which conditions does this method work?

Definition [Niqui] A digit set $\varphi_1, \dots, \varphi_k : I \to I$ is admissible if

- $\bigcap_{i=0}^{\infty} \varphi_{d_0}(\dots \varphi_{d_i}(I) \dots)$ is always a singleton
- $\bigcup_{j=0}^{k} \varphi_j(I^o) = I^o$.

(I^o is the interior of I)

Admissible digit sets and Möbius maps

Theorem [Niqui] An admissible digit set yields an admissible representation.

Theorem [Niqui] An admissible digit set yields an admissible representation.
There is still a choice which functions $\varphi_1, \ldots, \varphi_k : I \to I$ we take.
A popular way to choose them is via Möbius maps, maps of the shape

$$\frac{ax+b}{cx+d}$$

All the maps we have seen are Möbius maps.
They are usually viewed as matrices

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Möbius maps

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Möbius maps $\varphi_0, \ldots, \varphi_k$ give a very concrete way of computing with infinite sequences. Requirements:

- non-singular: $ad - bc \neq 0$
- increasing: $x < y \rightarrow \varphi(x) < \varphi(y)$
- refining: $\varphi(I) \subseteq I$
- shrinking: $\bigcap_{i=0}^{\infty} \varphi_{d_0}(\ldots \varphi_{d_i}(I) \ldots)$ is a point,
- maximal $\bigcup_{j=0}^{k} \varphi_j(I^o) = I^o$.

Stern Brocot Möbius maps

$$L(x) = \frac{x}{x+1}, R(x) = x+1, M(x) = \frac{2x+1}{x+2}$$

$$L = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, R = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, M = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Stern Brocot Möbius maps

$$L(x) = \frac{x}{x+1}, R(x) = x+1, M(x) = \frac{2x+1}{x+2}$$

$$L = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, R = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, M = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Now we can define a function as another Möbius map, for example $D(x) = 2x$:

$$D = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

Stern Brocot Möbius maps

$$L(x) = \frac{x}{x+1}, R(x) = x + 1, M(x) = \frac{2x+1}{x+2}$$

$$L = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, R = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, M = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Now we can define a function as another Möbius map, for example $D(x) = 2x$:

$$D = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

An actual algorithm has to compute output (digits) from Ds, with $s \in \{L, R, M\}^\omega$

Stern Brocot Möbius maps

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix} =?$$

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 4 & 2 \\ 1 & 2 \end{pmatrix} =?$$

Stern Brocot Möbius maps

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix} =?$$

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 4 & 2 \\ 1 & 2 \end{pmatrix} =?$$

- In case s starts with R, we can output R and continue. (Note that the function D has changed.)
- In the other cases we (may) have to consume more input first