Admissible Digit Sets

Jesse Hughes[1,2] Milad Niqui[2]

[1]Technical University of Eindhoven

[2]Radboud University of Nijmegen

September 28, 2004

Outline

Outline

Outline

1. Digit sets
   - Binary representation
   - Möbius maps and digit sets
   - The Stern-Brocot representation
2. Admissibility
   - Admissible digit sets
   - The homographic algorithm

Outline

1. Digit sets
   - Binary representation
   - Möbius maps and digit sets
   - The Stern-Brocot representation

2. Admissibility
   - Admissible digit sets
   - The homographic algorithm

Outline

1. Digit sets
   - Binary representation
   - Möbius maps and digit sets
   - The Stern-Brocot representation

2. Admissibility
   - Admissible digit sets
   - The homographic algorithm

Outline

1. Digit sets
   - Binary representation
   - Möbius maps and digit sets
   - The Stern-Brocot representation
2. Admissibility
   - Admissible digit sets
   - The homographic algorithm

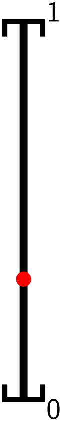
Outline

1. Digit sets
   - Binary representation
   - Möbius maps and digit sets
   - The Stern-Brocot representation
2. Admissibility
   - Admissible digit sets
   - The homographic algorithm

Outline

1. Digit sets
   - Binary representation
   - Möbius maps and digit sets
   - The Stern-Brocot representation
2. Admissibility
   - Admissible digit sets
   - The homographic algorithm

The standard binary representation of $[0, 1]$.



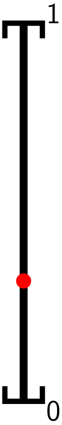
Think of binary representations in $[0, 1]$, like

$$0.010100010\ldots$$

$$\{0, 1\}^\omega \longrightarrow [0, 1]$$

$$x_1 x_2 x_3 \ldots \longmapsto \sum_{i=0}^{\infty} x_i \cdot 2^{-i}$$

The standard binary representation of $[0,1]$.



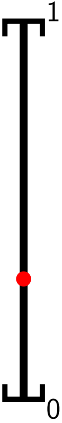
Think of binary representations in $[0,1]$, like

$$0.010100010\ldots$$

$$\{0,1\}^{\omega} \longrightarrow [0,1]$$

$$x_1\, x_2\, x_3\, \ldots \longmapsto \sum_{i=0}^{\infty} x_i \cdot 2^{-i}$$

The standard binary representation of $[0, 1]$.



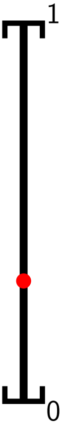
Think of binary representations in $[0, 1]$, like

$$0.010100010\dots$$

$$\{0,1\}^\omega \longrightarrow [0,1]$$

$$x_1\, x_2\, x_3\, \dots \longmapsto \sum_{i=0}^{\infty} x_i \cdot 2^{-i}$$

The standard binary representation of $[0, 1]$.

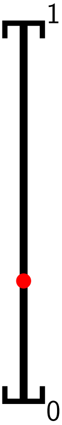


Think: receiving one bit at a time.

Each bit restricts the set of possibilities.

- With 0 bits, x could be anything in $[0, 1]$.
- When we see "0.0", the options are reduced.
- "0.01" reduces them further.

The standard binary representation of $[0, 1]$.

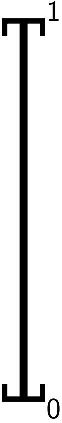


Think: receiving one bit at a time.

Each bit restricts the set of possibilities.

- With 0 bits, x could be anything in $[0, 1]$.
- When we see "0.0", the options are reduced.
- "0.01" reduces them further.

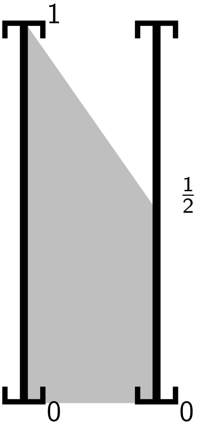
The standard binary representation of $[0, 1]$.



Think: receiving one bit at a time.
Each bit restricts the set of possibilities.

- With 0 bits, x could be anything in $[0, 1]$.
- When we see "0.0", the options are reduced.
- "0.01" reduces them further.

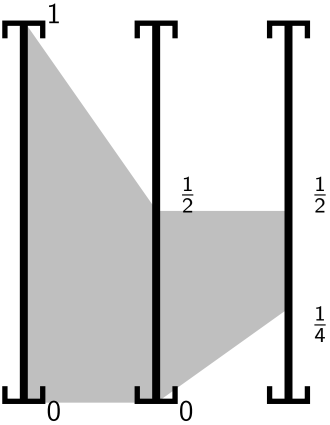
The standard binary representation of $[0, 1]$.



Think: receiving one bit at a time.
Each bit restricts the set of possibilities.

- With 0 bits, x could be anything in $[0, 1]$.
- When we see “0.0”, the options are reduced.
- “0.01” reduces them further.

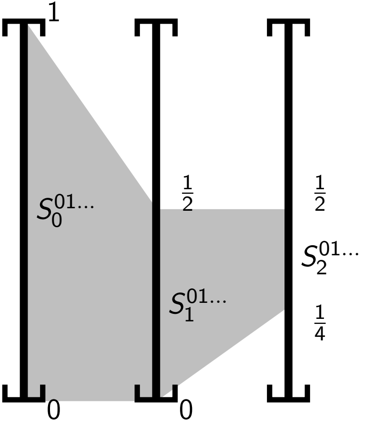
The standard binary representation of $[0, 1]$.



Think: receiving one bit at a time.
Each bit restricts the set of possibilities.

- With 0 bits, x could be anything in $[0, 1]$.
- When we see "0.0", the options are reduced.
- "0.01" reduces them further.

The standard binary representation of $[0, 1]$.



$$\vec{x} = x_1\, x_2\, x_3\ \ldots$$

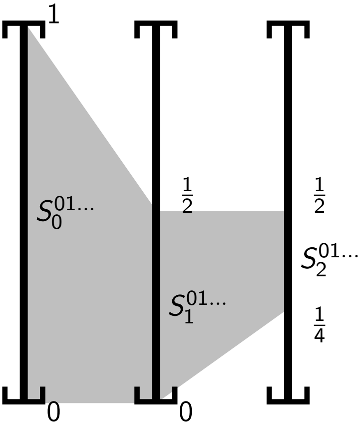
$$S_0^{\vec{x}} \supsetneq S_1^{\vec{x}} \supsetneq S_2^{\vec{x}} \supsetneq S_3^{\vec{x}} \supsetneq \ldots$$

Some features:

- $\bigcap S_i^{\vec{x}}$ is a singleton.
- For each x, there is a sequence $\vec{x}$ such that $\bigcap S_i^{\vec{x}} = \{x\}$.

Each sequence represents some x and each x is represented.

The standard binary representation of $[0,1]$.



$$\vec{x} = x_1\, x_2\, x_3\ \ldots$$

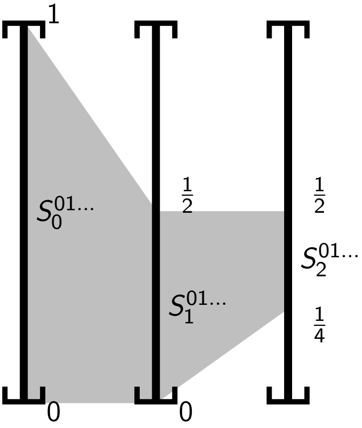
$$S_0^{\vec{x}} \supsetneq S_1^{\vec{x}} \supsetneq S_2^{\vec{x}} \supsetneq S_3^{\vec{x}} \supsetneq \ldots$$

Some features:

- $\bigcap S_i^{\vec{x}}$ is a singleton.
- For each x, there is a sequence $\vec{x}$ such that $\bigcap S_i^{\vec{x}} = \{x\}$.

Each sequence represents some x and each x is represented.

The standard binary representation of $[0, 1]$.



$$\vec{x} = x_1\, x_2\, x_3\, \ldots$$

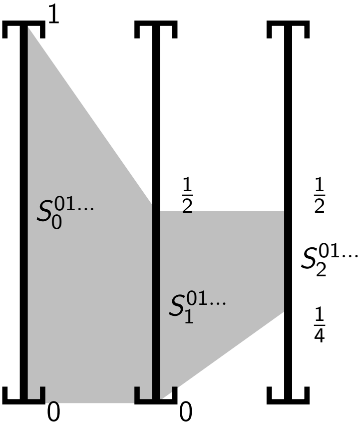
$$S_0^{\vec{x}} \supsetneq S_1^{\vec{x}} \supsetneq S_2^{\vec{x}} \supsetneq S_3^{\vec{x}} \supsetneq \ldots$$

Some features:

- $\bigcap S_i^{\vec{x}}$ is a singleton.
- For each x, there is a sequence $\vec{x}$ such that $\bigcap S_i^{\vec{x}} = \{x\}$.

Each sequence represents some x and each x is represented.

The standard binary representation of $[0, 1]$.



$$\vec{x} = x_1\, x_2\, x_3\, \ldots$$

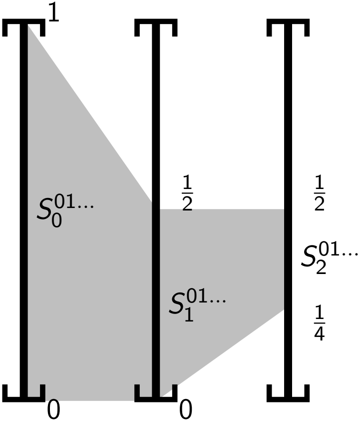
$$S_0^{\vec{x}} \supsetneq S_1^{\vec{x}} \supsetneq S_2^{\vec{x}} \supsetneq S_3^{\vec{x}} \supsetneq \ldots$$

Some features:

- $\bigcap S_i^{\vec{x}}$ is a singleton.
- For each x, there is a sequence $\vec{x}$ such that $\bigcap S_i^{\vec{x}} = \{x\}$.

Each sequence represents some x and each x is represented.

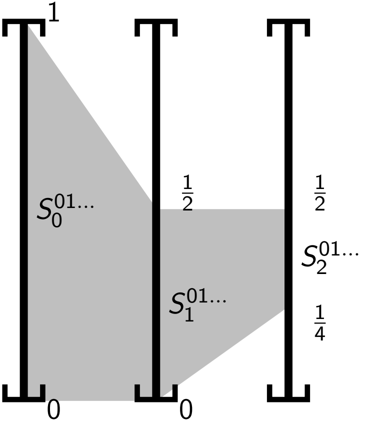
How to construct the sets $S_i^{\vec{x}}$



$$\phi_0(x) = \frac{x}{2}$$

$$\phi_1(x) = \frac{x+1}{2}$$

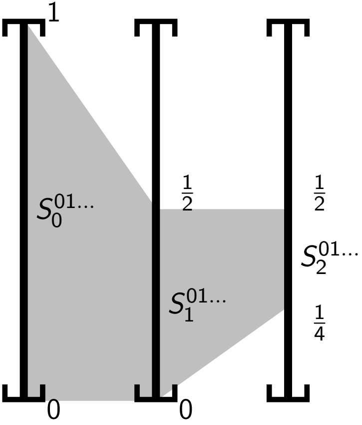
How to construct the sets $S_i^{\vec{x}}$



$$\phi_0(x) = \frac{x}{2}$$

$$\phi_1(x) = \frac{x+1}{2}$$

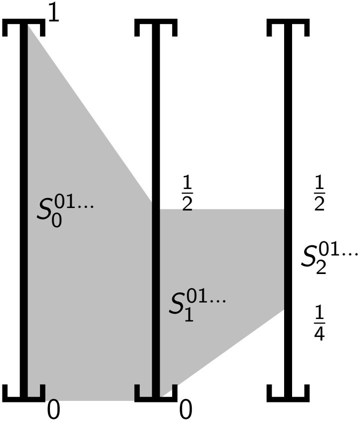
How to construct the sets $S_i^{\vec{x}}$



$$\phi_0(x) = \frac{x}{2}$$
$$\phi_1(x) = \frac{x+1}{2}$$

$$S_0^{01\ldots} = [0,1]$$
$$S_1^{01\ldots} = \phi_0([0,1])$$
$$S_2^{01\ldots} = \phi_0 \circ \phi_1([0,1])$$
$$S_i^{\vec{x}} = \phi_{x_0} \circ \ldots \circ \phi_{x_i}([0,1])$$
$$\bigcap_{i \in \mathbb{N}} S_i^{\vec{x}} = \{0.x_0\, x_1 \ldots\}$$

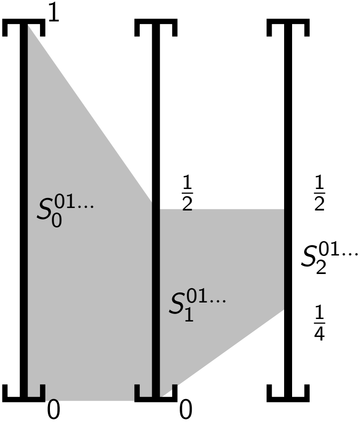
How to construct the sets $S_i^{\vec{x}}$



$$\phi_0(x) = \frac{x}{2}$$
$$\phi_1(x) = \frac{x+1}{2}$$

$$S_0^{01\dots} = [0,1]$$
$$S_1^{01\dots} = \phi_0([0,1])$$
$$S_2^{01\dots} = \phi_0 \circ \phi_1([0,1])$$
$$S_i^{\vec{x}} = \phi_{x_0} \circ \dots \circ \phi_{x_i}([0,1])$$
$$\bigcap_{i \in \mathbb{N}} S_i^{\vec{x}} = \{0.x_0\, x_1\, \dots\}$$

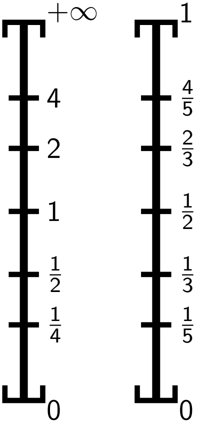
How to construct the sets $S_i^{\vec{x}}$



$$\phi_0(x) = \frac{x}{2}$$
$$\phi_1(x) = \frac{x+1}{2}$$

$$S_0^{01\ldots} = [0,1]$$
$$S_1^{01\ldots} = \phi_0([0,1])$$
$$S_2^{01\ldots} = \phi_0 \circ \phi_1([0,1])$$
$$S_i^{\vec{x}} = \phi_{x_0} \circ \ldots \circ \phi_{x_i}([0,1])$$
$$\bigcap_{i \in \mathbb{N}} S_i^{\vec{x}} = \{0.x_0\, x_1\, \ldots\}$$

1, $+\infty$, what's the difference?

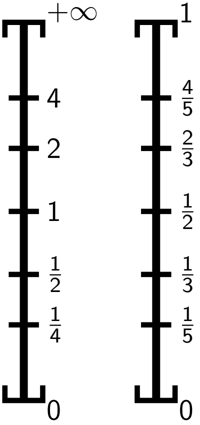


Work with $[0, +\infty]$ or $[0, 1]$?

The choice is arbitrary.

Squint and you can't tell the difference.

1, $+\infty$, what's the difference?

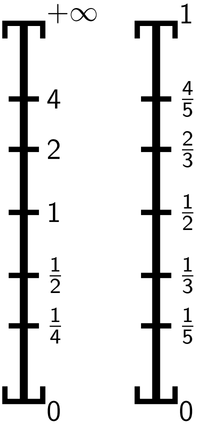


Work with $[0, +\infty]$ or $[0, 1]$?

The choice is arbitrary.

Squint and you can't tell the difference.

1, $+\infty$, what's the difference?



Work with $[0, +\infty]$ or $[0, 1]$?

The choice is arbitrary.

Squint and you can't tell the difference.

Möbius maps

Recall $\phi_0(x) = \frac{x}{2}, \qquad \phi_1(x) = \frac{x+1}{2}$.

Möbius map: a function

$$A(x) = \frac{ax+b}{cx+d}$$

where $a, b, c, d \in \mathbb{R}$.

We are interested in Möbius maps that are

- strictly increasing,
- refining $(A : [0, +\infty] \to [0, +\infty])$.

Möbius maps

Recall $\phi_0(x) = \frac{x}{2}, \qquad \phi_1(x) = \frac{x+1}{2}$.
Möbius map: a function

$$A(x) = \frac{ax + b}{cx + d}$$

where $a, b, c, d \in \mathbb{R}$.
We are interested in Möbius maps that are

- strictly increasing,
- refining ($A : [0, +\infty] \to [0, +\infty]$).

Möbius maps

Recall $\phi_0(x) = \frac{x}{2}$, $\qquad \phi_1(x) = \frac{x+1}{2}$.
Möbius map: a function

$$A(x) = \frac{ax + b}{cx + d}$$

where $a, b, c, d \in \mathbb{R}$.
We are interested in Möbius maps that are

- strictly increasing,
- refining ($A : [0, +\infty] \to [0, +\infty]$).

Möbius maps

Recall $\phi_0(x) = \frac{x}{2}$, $\phi_1(x) = \frac{x+1}{2}$.
Möbius map: a function

$$A(x) = \frac{ax + b}{cx + d}$$

where $a, b, c, d \in \mathbb{R}$.
We are interested in Möbius maps that are

- strictly increasing,
- refining ($A : [0, +\infty] \to [0, +\infty]$).

Digit sets

Möbius maps are our *digits*.

Let $\Phi = \{\phi_0, \ldots, \phi_k\}$ be a set of Möbius maps.

A sequence $\vec{x} = \phi_{i_0}\, \phi_{i_1}\, \phi_{i_2}\, \ldots$ *represents* x if

$$\bigcap_{n=0}^{\infty} \underbrace{\phi_{i_0} \circ \phi_{i_1} \circ \ldots \circ \phi_{i_n}([0, +\infty])}_{S_n^{\vec{x}}} = \{x\}.$$

Φ is a *digit set* if each x is represented.

Digit sets

Möbius maps are our *digits*.

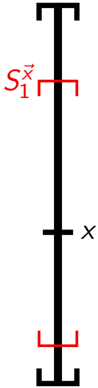
Let $\Phi = \{\phi_0, \dots, \phi_k\}$ be a set of Möbius maps.

A sequence $\vec{x} = \phi_{i_0}\, \phi_{i_1}\, \phi_{i_2}\, \dots$ *represents* x if

$$\bigcap_{n=0}^{\infty} \underbrace{\phi_{i_0} \circ \phi_{i_1} \circ \dots \circ \phi_{i_n}([0, +\infty])}_{S_n^{\vec{x}}} = \{x\}.$$

Φ is a *digit set* if each x is represented.

Digit sets

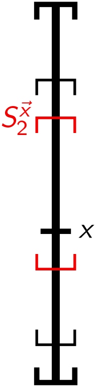


Möbius maps are our *digits*.
Let $\Phi = \{\phi_0, \ldots, \phi_k\}$ be a set of Möbius maps.
A sequence $\vec{x} = \phi_{i_0}\,\phi_{i_1}\,\phi_{i_2}\,\ldots$ *represents* x if

$$\bigcap_{n=0}^{\infty} \underbrace{\phi_{i_0} \circ \phi_{i_1} \circ \ldots \circ \phi_{i_n}([0, +\infty])}_{S_n^{\vec{x}}} = \{x\}.$$

Φ is a *digit set* if each x is represented.

Digit sets

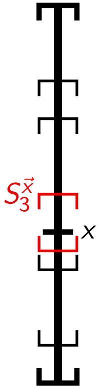


Möbius maps are our *digits*.
Let $\Phi = \{\phi_0, \ldots, \phi_k\}$ be a set of Möbius maps.
A sequence $\vec{x} = \phi_{i_0}\, \phi_{i_1}\, \phi_{i_2}\, \ldots$ *represents* x if

$$\bigcap_{n=0}^{\infty} \underbrace{\phi_{i_0} \circ \phi_{i_1} \circ \ldots \circ \phi_{i_n}([0, +\infty])}_{S_n^{\vec{x}}} = \{x\}.$$

Φ is a *digit set* if each x is represented.

Digit sets

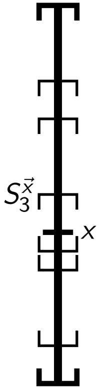


Möbius maps are our *digits*.
Let $\Phi = \{\phi_0, \ldots, \phi_k\}$ be a set of Möbius maps.
A sequence $\vec{x} = \phi_{i_0}\, \phi_{i_1}\, \phi_{i_2}\, \ldots$ *represents* x if

$$\bigcap_{n=0}^{\infty} \underbrace{\phi_{i_0} \circ \phi_{i_1} \circ \ldots \circ \phi_{i_n}([0, +\infty])}_{S_n^{\vec{x}}} = \{x\}.$$

Φ is a *digit set* if each x is represented.

Digit sets



Möbius maps are our *digits*.
Let $\Phi = \{\phi_0, \ldots, \phi_k\}$ be a set of Möbius maps.
A sequence $\vec{x} = \phi_{i_0}\, \phi_{i_1}\, \phi_{i_2}\, \ldots$ *represents* x if

$$\bigcap_{n=0}^{\infty} \underbrace{\phi_{i_0} \circ \phi_{i_1} \circ \ldots \circ \phi_{i_n}([0, +\infty])}_{S_n^{\vec{x}}} = \{x\}.$$

Φ is a *digit set* if each x is represented.

Good digit sets

Φ is a *good digit set* if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i([0,+\infty])$ cover $[0,+\infty]$.

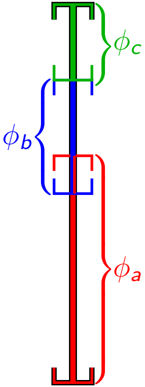
Theorem

Good digit sets are digit sets.

Good digit sets yield a *total representation*, i.e. $\Phi^\omega \to [0,+\infty]$ is

- total
-
-

Good digit sets



Φ is a *good digit set* if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

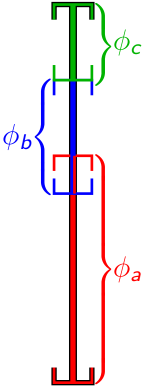
Theorem

Good digit sets are digit sets.

Good digit sets yield a *total representation*, i.e. $\Phi^\omega \to [0, +\infty]$ is

- total,
- continuous
- surjective

Good digit sets



Φ is a *good digit set* if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

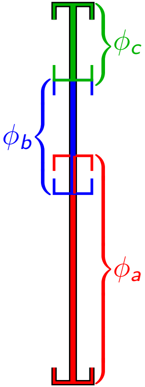
Theorem

Good digit sets are digit sets.

Good digit sets yield a *total representation*, i.e. $\Phi^\omega \to [0, +\infty]$ is

- total,
- continuous,
- surjective.

Good digit sets



Φ is a *good digit set* if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

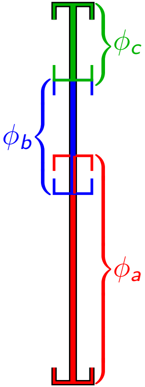
Theorem

Good digit sets are digit sets.

Good digit sets yield a *total representation*, i.e. $\Phi^\omega \to [0, +\infty]$ is

- total,
- continuous,
- surjective.

Good digit sets



Φ is a *good digit set* if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

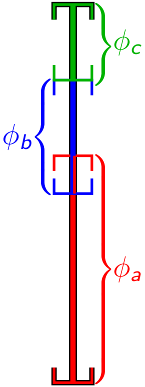
Theorem

Good digit sets are digit sets.

Good digit sets yield a *total representation*, i.e. $\Phi^\omega \to [0, +\infty]$ is

- total,
- continuous,
- surjective.

Good digit sets



Φ is a *good digit set* if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

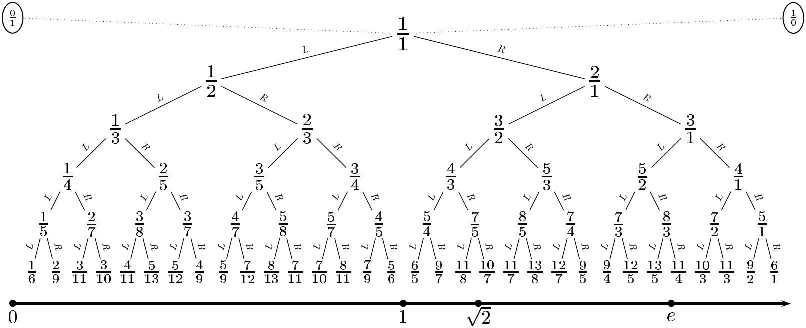
Theorem

Good digit sets are digit sets.

Good digit sets yield a *total representation*, i.e. $\Phi^\omega \to [0, +\infty]$ is

- total,
- continuous,
- surjective.

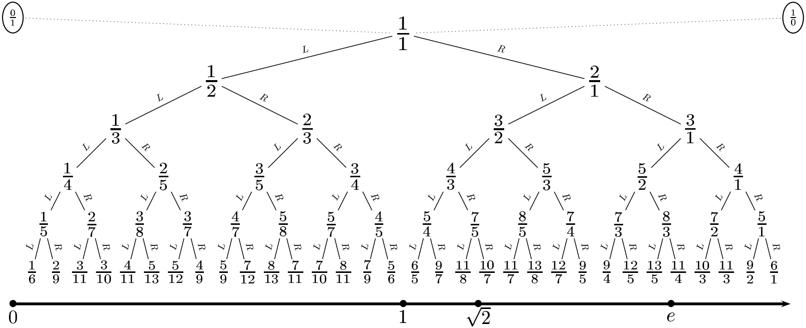
The Stern-Brocot representation is a digit set



How to make the tree:

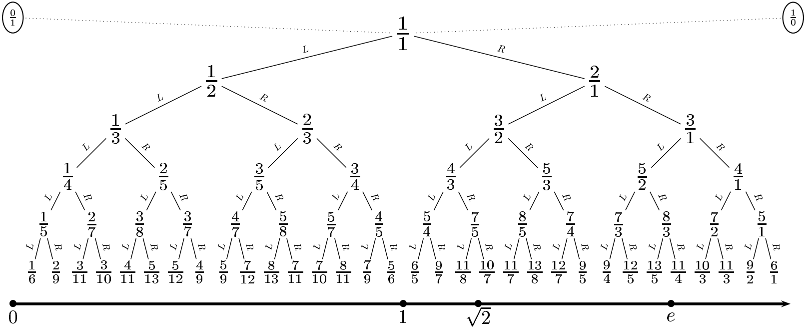
If my parents are $\frac{a}{b}$ and $\frac{c}{d}$, then I am $\frac{a+c}{b+d}$.

The Stern-Brocot representation is a digit set



The Stern-Brocot representation maps finite sequences of $\{L, R\}$ to rationals.

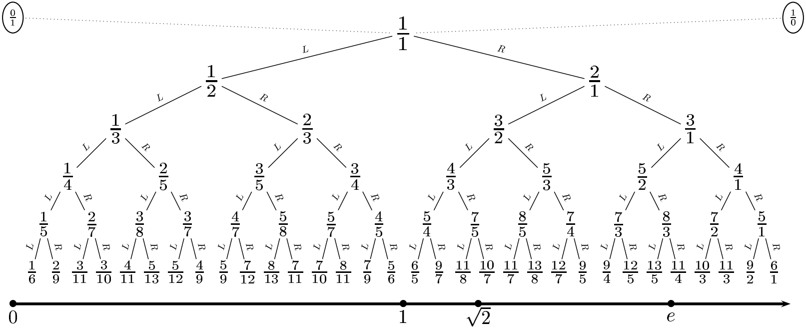
The Stern-Brocot representation is a digit set



The Stern-Brocot representation maps finite sequences of $\{L, R\}$ to rationals.

Easy to show: infinite sequences yield Cauchy sequences of rationals.

The Stern-Brocot representation is a digit set

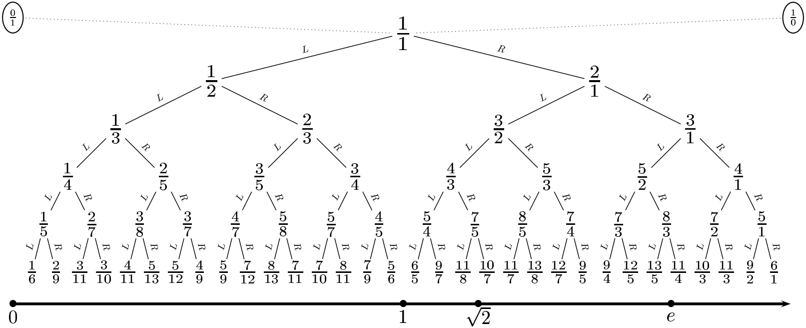


The Stern-Brocot representation maps finite sequences of $\{L, R\}$ to rationals.

Easy to show: infinite sequences yield Cauchy sequences of rationals.

Careful with that metric!

The Stern-Brocot representation is a digit set



$$L \ldots \in [0, 1]$$

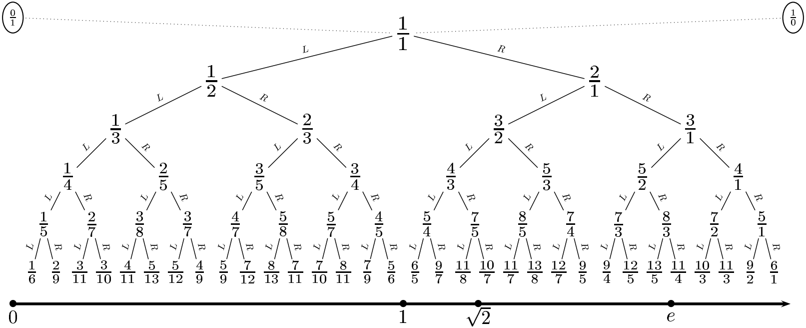
$$LR \ldots \in [\frac{1}{2}, 1]$$

$$LRR \ldots \in [\frac{2}{3}, 1]$$

A nested sequence of sets $S_i^{\vec{x}}$.

Each $S_i^{\vec{x}}$ is bounded by the parents of $x_1\, x_2\, \ldots\, x_i$.

The Stern-Brocot representation is a digit set



$$L \ldots \in [0, 1]$$

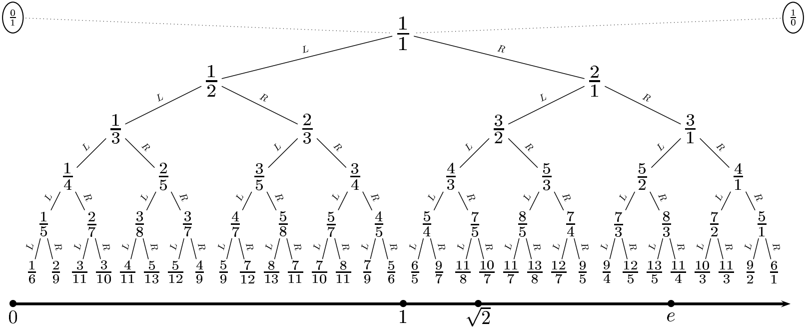
$$LR \ldots \in [\frac{1}{2}, 1]$$

$$LRR \ldots \in [\frac{2}{3}, 1]$$

A nested sequence of sets $S_i^{\vec{x}}$.

Each $S_i^{\vec{x}}$ is bounded by the parents of $x_1 x_2 \ldots x_i$.

The Stern-Brocot representation is a digit set



$$L \ldots \in [0, 1]$$

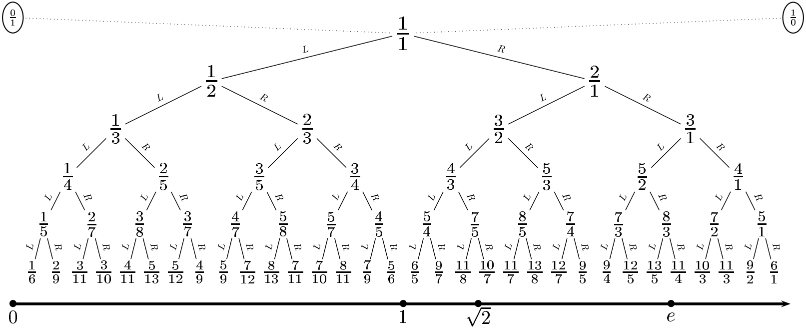
$$LR \ldots \in [\frac{1}{2}, 1]$$

$$LRR \ldots \in [\frac{2}{3}, 1]$$

A nested sequence of sets $S_i^{\vec{x}}$.

Each $S_i^{\vec{x}}$ is bounded by the parents of $x_1\, x_2\, \ldots\, x_i$.

The Stern-Brocot representation is a digit set



$$\phi_L(x) = \frac{x}{x+1} \qquad \phi_R(x) = x + 1$$

$\{\phi_L, \phi_R\}$ is a good digit set.

Outline

1. Digit sets
   - Binary representation
   - Möbius maps and digit sets
   - The Stern-Brocot representation
2. Admissibility
   - Admissible digit sets
   - The homographic algorithm

Φ-Computability

Let Φ be a good digit set.

$f : [0, +\infty] \to [0, +\infty]$ is Φ-computable iff f has a continuous Φ^ω lifting.

$$\begin{array}{ccc} \Phi^\omega & \overset{f^\sharp}{\dashrightarrow} & \Phi^\omega \\ \downarrow & & \downarrow \\ [0, +\infty] & \underset{f}{\longrightarrow} & [0, +\infty] \end{array}$$

Good digit sets aren't *very* good.

$x \mapsto 2x$ isn't Stern-Brocot-computable.

Φ-Computability

Let Φ be a good digit set.
$f : [0, +\infty] \to [0, +\infty]$ is Φ-computable iff f has a continuous Φ^ω lifting.

$$\begin{array}{ccc} \Phi^\omega & \overset{f^\sharp}{\dashrightarrow} & \Phi^\omega \\ \downarrow & & \downarrow \\ [0, +\infty] & \underset{f}{\longrightarrow} & [0, +\infty] \end{array}$$

Good digit sets aren't *very* good.
$x \mapsto 2x$ isn't Stern-Brocot-computable.

Φ-Computability

Let Φ be a good digit set.
$f : [0, +\infty] \to [0, +\infty]$ is Φ-computable iff f has a continuous Φ^ω lifting.

$$\begin{array}{ccc} \Phi^\omega & \overset{f^\sharp}{\dashrightarrow} & \Phi^\omega \\ \downarrow & & \downarrow \\ [0, +\infty] & \underset{f}{\longrightarrow} & [0, +\infty] \end{array}$$

Good digit sets aren't *very* good.
$x \mapsto 2x$ isn't Stern-Brocot-computable.

Admissible representations

$p : \Phi^\omega \to [0, +\infty]$ is an *admissible representation* if it is

- continuous,
- surjective,
- maximal, i.e. for every continuous r:

If $\Phi^\omega \to [0, +\infty]$ is admissible, any continuous f is Φ-computable.

Admissible representations

$p : \Phi^\omega \to [0, +\infty]$ is an *admissible representation* if it is

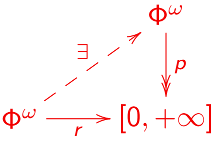
- continuous,
- surjective,
- maximal, i.e. for every continuous r:

If $\Phi^\omega \to [0, +\infty]$ is admissible, any continuous f is Φ-computable.

Admissible representations

$p : \Phi^\omega \to [0, +\infty]$ is an *admissible representation* if it is

- continuous,
- surjective,
- maximal, i.e. for every continuous r:

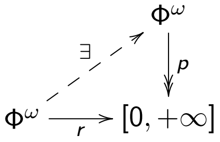


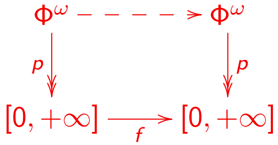
If $\Phi^\omega \to [0, +\infty]$ is admissible, any continuous f is Φ-computable.

Admissible representations

$p : \Phi^\omega \to [0, +\infty]$ is an *admissible representation* if it is

- continuous,
- surjective,
- maximal, i.e. for every continuous r:





If $\Phi^\omega \to [0, +\infty]$ is admissible, any continuous f is Φ-computable.

Admissible digit sets

Φ is an *admissible digit set* (ADS) if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i((0, +\infty))$ cover $(0, +\infty)$.

(2) replaces

- The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

for good digit sets.

Theorem

Admissible digit sets yield admissible representations.

Not ADS!

Admissible digit sets

Φ is an *admissible digit set* (ADS) if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i((0, +\infty))$ cover $(0, +\infty)$.

(2) replaces

- The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

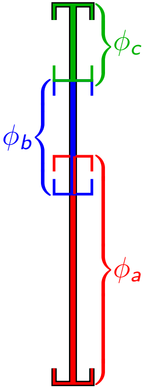
for good digit sets.

Theorem

Admissible digit sets yield admissible representations.

Not ADS!

Admissible digit sets



Not ADS!

Φ is an *admissible digit set* (ADS) if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i((0, +\infty))$ cover $(0, +\infty)$.

(2) replaces

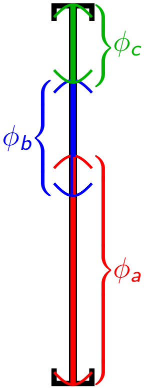
- The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

for good digit sets.

Theorem

Admissible digit sets yield admissible representations.

Admissible digit sets



Not ADS!

Φ is an *admissible digit set* (ADS) if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i((0, +\infty))$ cover $(0, +\infty)$.

(2) replaces

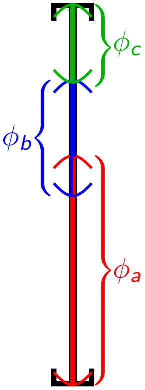
- The sets $\phi_i([0, +\infty])$ cover $[0, +\infty]$.

for good digit sets.

Theorem

Admissible digit sets yield admissible representations.

Admissible digit sets



Not ADS!

Φ is an *admissible digit set* (ADS) if

1. Loosely: $\bigcap S_i^{\vec{x}}$ is always a singleton.
2. The sets $\phi_i((0,+\infty))$ cover $(0,+\infty)$.

(2) replaces

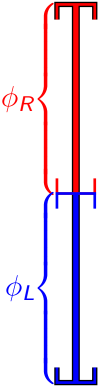
- The sets $\phi_i([0,+\infty])$ cover $[0,+\infty]$.

for good digit sets.

Theorem

Admissible digit sets yield admissible representations.

The Stern-Brocot representation is not ADS

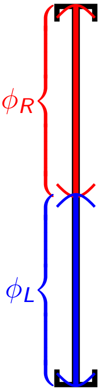


S-B is a good digit set...
but not an admissible digit set.

$$\phi_L([0,+\infty]) = [0,1]$$
$$\phi_R([0,+\infty]) = [1,+\infty]$$

Solution: Add $\phi_M(x) = \frac{2x+1}{x+2}$.

The Stern-Brocot representation is not ADS

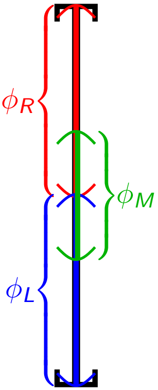


S-B is a good digit set...
but not an admissible digit set.

$$\phi_L((0,+\infty)) = (0,1)$$
$$\phi_R((0,+\infty)) = (1,+\infty)$$

Solution: Add $\phi_M(x) = \frac{2x+1}{x+2}$.

The Stern-Brocot representation is not ADS



S-B is a good digit set. . .
but not an admissible digit set.

$$\phi_L((0,+\infty)) = (0,1)$$
$$\phi_R((0,+\infty)) = (1,+\infty)$$

Solution: Add $\phi_M(x) = \frac{2x+1}{x+2}$.

Why this subsection doesn't matter.

Let Φ be an ADS.
Aim: Construct an algorithm $H(A, -)$ computing Möbius maps A.

But $\Phi^\omega \to [0, +\infty]$ is an admissible representation.

Any continuous $f : [0, +\infty] \to [0, +\infty]$ lifts to Φ^ω.

$$\begin{array}{ccc} \Phi^\omega & \dashrightarrow & \Phi^\omega \\ \downarrow p & & \downarrow p \\ [0, +\infty] & \xrightarrow[f]{} & [0, +\infty] \end{array}$$

But formal verifications require explicit algorithms.

Why this subsection doesn't matter.

Let Φ be an ADS.
Aim: Construct an algorithm $H(A, -)$ computing Möbius maps A.

But $\Phi^\omega \to [0, +\infty]$ is an admissible representation.

Any continuous $f : [0, +\infty] \to [0, +\infty]$ lifts to Φ^ω.

$$\begin{array}{ccc} \Phi^\omega & \dashrightarrow & \Phi^\omega \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle p} \\ [0, +\infty] & \xrightarrow[f]{} & [0, +\infty] \end{array}$$

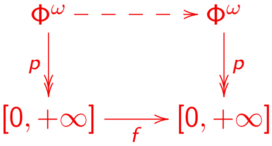
But formal verifications require explicit algorithms.

Why this subsection doesn't matter.

Let Φ be an ADS.
Aim: Construct an algorithm $H(A, -)$ computing Möbius maps A.

But $\Phi^\omega \to [0, +\infty]$ is an admissible representation.

Any continuous $f : [0, +\infty] \to [0, +\infty]$ lifts to Φ^ω.



But formal verifications require explicit algorithms.

Why this subsection does matter.

Let Φ be an ADS.
Aim: Construct an algorithm $H(A, -)$ computing Möbius maps A.

But $\Phi^\omega \to [0, +\infty]$ is an admissible representation.

Any continuous $f : [0, +\infty] \to [0, +\infty]$ lifts to Φ^ω.

$$\begin{array}{ccc} \Phi^\omega & \dashrightarrow & \Phi^\omega \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle p} \\ [0, +\infty] & \xrightarrow[f]{} & [0, +\infty] \end{array}$$

But formal verifications require explicit algorithms.

An algorithm for computing Möbius maps

Let $\mathbb{M}$ be the set of refining Möbius maps.

We explicitly defined $H : \mathbb{M} \times \Phi^\omega \to \Phi^\omega$ so that

$$\begin{array}{ccc} \Phi^\omega & \xrightarrow{H(A,-)} & \Phi^\omega \\ \downarrow p & & \downarrow p \\ [0, +\infty] & \xrightarrow[A]{} & [0, +\infty] \end{array}$$

H is the *homographic algorithm.*

An algorithm for computing Möbius maps

Let $\mathbb{M}$ be the set of refining Möbius maps.

We explicitly defined $H : \mathbb{M} \times \Phi^\omega \to \Phi^\omega$ so that

$$\begin{array}{ccc} \Phi^\omega & \xrightarrow{H(A,-)} & \Phi^\omega \\ \downarrow p & & \downarrow p \\ [0,+\infty] & \xrightarrow[A]{} & [0,+\infty] \end{array}$$

H is the *homographic algorithm*.

An algorithm for computing Möbius maps

Let $\mathbb{M}$ be the set of refining Möbius maps.

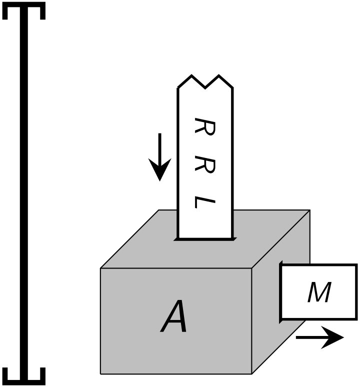
We explicitly defined $H : \mathbb{M} \times \Phi^\omega \to \Phi^\omega$ so that

$$\begin{array}{ccc} \Phi^\omega & \xrightarrow{H(A,-)} & \Phi^\omega \\ \Big\downarrow p & & \Big\downarrow p \\ [0,+\infty] & \xrightarrow[A]{} & [0,+\infty] \end{array}$$

H is the *homographic algorithm*.

The very (very) rough idea behind the algorithm
(but with pictures)

H is the *homographic algorithm*.

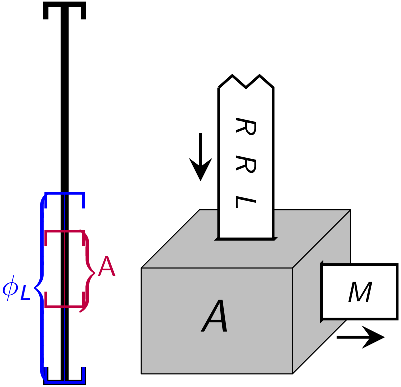


Least fixed point construction that

- outputs a digit when possible or
- absorbs more input when needed.

The very (very) rough idea behind the algorithm
(but with pictures)

H is the *homographic algorithm*.

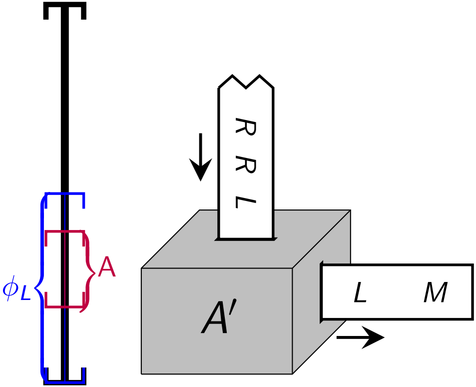


Least fixed point construction that

- outputs a digit when possible or
- absorbs more input when needed.

The very (very) rough idea behind the algorithm
(but with pictures)

H is the *homographic algorithm*.

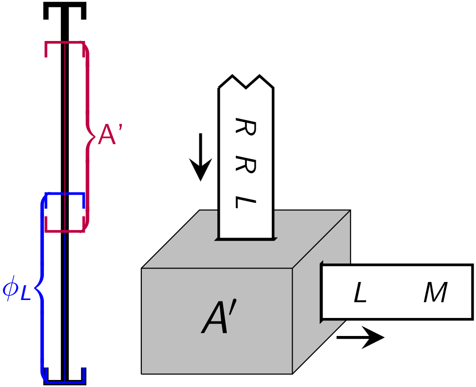


Least fixed point construction that

- outputs a digit when possible or
- absorbs more input when needed.

The very (very) rough idea behind the algorithm
(but with pictures)

H is the *homographic algorithm*.

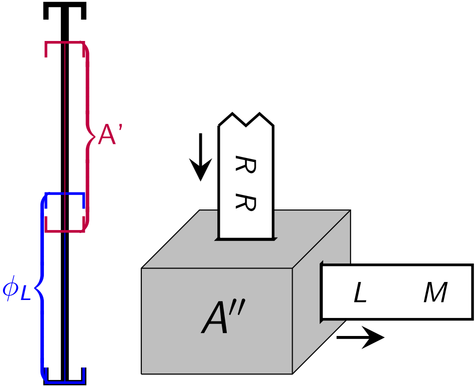


Least fixed point construction that

- outputs a digit when possible or
- absorbs more input when needed.

The very (very) rough idea behind the algorithm
(but with pictures)

H is the *homographic algorithm*.



Least fixed point construction that

- outputs a digit when possible or
- absorbs more input when needed.

Er, so what did we do?

- Aim: investigate representations via Möbius maps
- Found sufficient conditions for
  - total representations
  - total, admissible representations
- modified Stern-Brocot to do formal arithmetic
- explicitly computed homographic algorithm for ADS

Er, so what did we do?

- Aim: investigate representations via Möbius maps
- Found sufficient conditions for
  - total representations
  - total, admissible representations
- modified Stern-Brocot to do formal arithmetic
- explicitly computed homographic algorithm for ADS

Er, so what did we do?

- Aim: investigate representations via Möbius maps
- Found sufficient conditions for
  - total representations
  - total, admissible representations
- modified Stern-Brocot to do formal arithmetic
- explicitly computed homographic algorithm for ADS

Er, so what did we do?

- Aim: investigate representations via Möbius maps
- Found sufficient conditions for
  - total representations
  - total, admissible representations
- modified Stern-Brocot to do formal arithmetic
- explicitly computed homographic algorithm for ADS

Er, so what did we do?

- Aim: investigate representations via Möbius maps
- Found sufficient conditions for
  - total representations
  - total, admissible representations
- modified Stern-Brocot to do formal arithmetic
- explicitly computed homographic algorithm for ADS

Er, so what did we do?

- Aim: investigate representations via Möbius maps
- Found sufficient conditions for
  - total representations
  - total, admissible representations
- modified Stern-Brocot to do formal arithmetic
- explicitly computed homographic algorithm for ADS

Outline

Möbius maps and matrices

$$A(x) = \frac{ax+b}{cx+d}$$

Same thing: A matrix $M_A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

Let $x, y \in [0, +\infty)$.

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{cases} \frac{x}{y} & \text{if } y \neq 0, \\ +\infty & \text{else.} \end{cases}$$

$$A(\frac{x}{y}) \text{``}=\text{''} M_A \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

Composition of Möbius maps is the same as multiplication of matrices.

Möbius maps and matrices

$$A(x) = \frac{ax + b}{cx + d}$$

Same thing: A matrix $M_A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

Let $x, y \in [0, +\infty)$.

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{cases} \frac{x}{y} & \text{if } y \neq 0, \\ +\infty & \text{else.} \end{cases}$$

$$A(\frac{x}{y}) \text{``}=\text{''} M_A \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

Composition of Möbius maps is the same as multiplication of matrices.

Möbius maps and matrices

$$A(x) = \frac{ax+b}{cx+d}$$

Same thing: A matrix $M_A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

Let $x, y \in [0, +\infty)$.

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{cases} \frac{x}{y} & \text{if } y \neq 0, \\ +\infty & \text{else.} \end{cases}$$

$$A(\frac{x}{y}) \text{``}=\text{''} M_A \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

Composition of Möbius maps is the same as multiplication of matrices.

Möbius maps and matrices

$$A(x) = \frac{ax+b}{cx+d}$$

Same thing: A matrix $M_A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

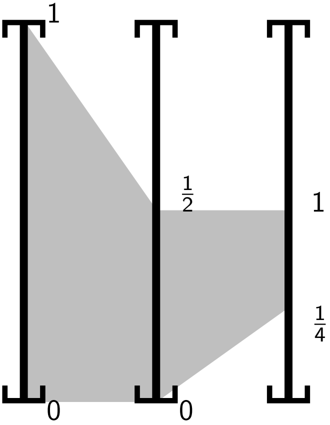
Let $x, y \in [0, +\infty)$.

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{cases} \frac{x}{y} & \text{if } y \neq 0, \\ +\infty & \text{else.} \end{cases}$$

$$A(\frac{x}{y}) \text{``}=\text{''} M_A \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

Composition of Möbius maps is the same as multiplication of matrices.

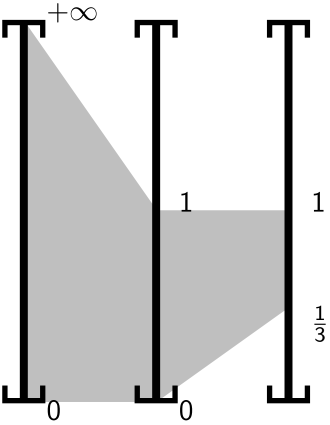
Translating ϕ_0, ϕ_1 to $[0, +\infty]$



$$\phi_0(x) = \frac{x}{2}$$

$$\phi_1(x) = \frac{x+1}{2}$$

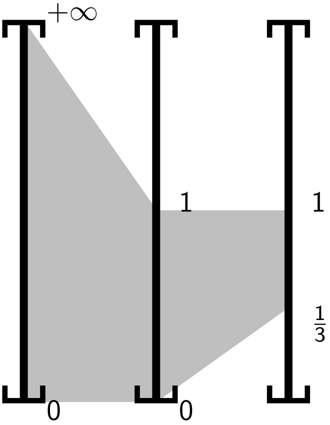
Translating ϕ_0, ϕ_1 to $[0, +\infty]$



$$\phi_0(x) = \frac{x}{2}$$

$$\phi_1(x) = \frac{x+1}{2}$$

Translating ϕ_0, ϕ_1 to $[0, +\infty]$



$$\phi_0(x) = \frac{x}{2}$$

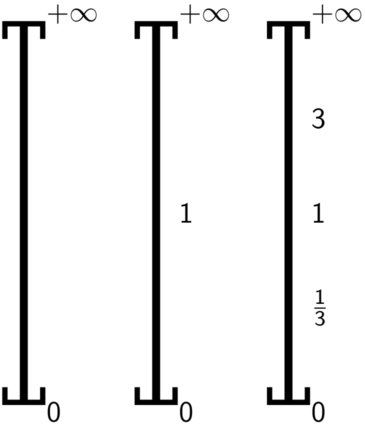
$$\phi_1(x) = \frac{x+1}{2}$$

$$[0, 1] = \phi_0([0, +\infty])$$

$$[\frac{1}{3}, 1] = \phi_0 \circ \phi_1([0, +\infty])$$

$$\frac{1}{\pi - 1} \text{ “} = \text{” } .010100010\ldots$$

Good digit sets have shrinking diameters.

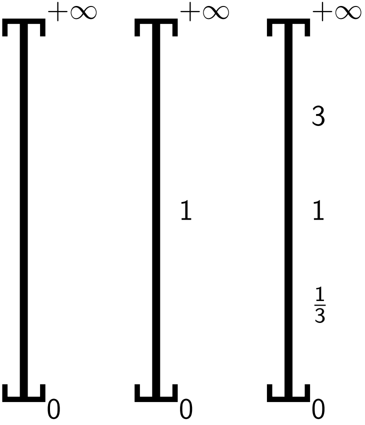


$$\phi_0(x) = \frac{x}{2}$$
$$\phi_1(x) = \frac{x+1}{2}$$

$[0, +\infty]$ inherits a metric from $[0, 1]$.

We use this metric to measure the "shrinking" of $\phi_{i_1}\phi_{i_2} \ldots \phi_{i_n}([0, +\infty])$.

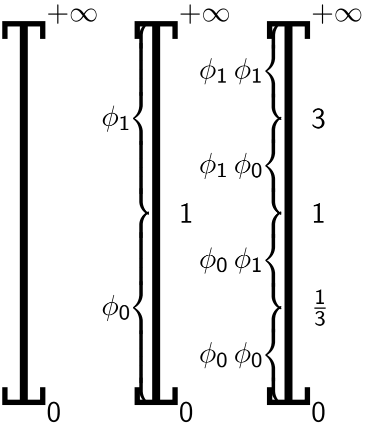
Good digit sets have shrinking diameters.



$$\phi_0(x) = \frac{x}{2}$$
$$\phi_1(x) = \frac{x+1}{2}$$

$[0, +\infty]$ inherits a metric from $[0, 1]$.
We use this metric to measure the "shrinking" of $\phi_{i_1}\phi_{i_2} \ldots \phi_{i_n}([0, +\infty])$.

Good digit sets have shrinking diameters.

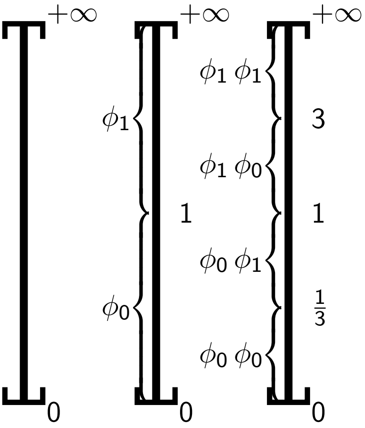


$\mathcal{B}(\Phi, n)$ measures the maximum diameter for n-length sequences.

$$\mathcal{B}(\Phi, 0) = 1$$
$$\mathcal{B}(\Phi, 1) = \frac{1}{2}$$
$$\mathcal{B}(\Phi, 2) = \frac{1}{4}$$

Good: $\lim_{j \to \infty} \mathcal{B}(\Phi, j) = 0$

Good digit sets have shrinking diameters.



$\mathcal{B}(\Phi, n)$ measures the maximum diameter for n-length sequences.

$$\mathcal{B}(\Phi, 0) = 1$$

$$\mathcal{B}(\Phi, 1) = \frac{1}{2}$$

$$\mathcal{B}(\Phi, 2) = \frac{1}{4}$$

Good: $\lim_{j\to\infty} \mathcal{B}(\Phi, j) = 0$

The rough idea behind the algorithm

When $A(x) \in \phi_j((0, +\infty))$ no matter what x is, output the digit ϕ_j.

Otherwise, absorb a digit from x to refine our calculation.

Define $A \sqsubseteq \phi_j \Leftrightarrow A([0, +\infty]) \subseteq \phi_j([0, +\infty])$.

$$H(A, \phi_{i_1}\phi_{i_2}\ldots) := \begin{cases} \phi_0\ H(\phi_0^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{if } A \sqsubseteq \phi_0 \\ \phi_1\ H(\phi_1^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_1 \\ \vdots & \\ \phi_k\ H(\phi_k^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_k \\ H(A \circ \phi_i, \phi_{i_2}\phi_{i_3}\ldots) & \textbf{otherwise}. \end{cases}$$

The rough idea behind the algorithm

When $A(x) \in \phi_j((0,+\infty))$ no matter what x is, output the digit ϕ_j.

Otherwise, absorb a digit from x to refine our calculation.

Define $A \sqsubseteq \phi_j \Leftrightarrow A([0,+\infty]) \subseteq \phi_j([0,+\infty])$.

$$H(A, \phi_{i_1}\phi_{i_2}\ldots) := \begin{cases} \phi_0\ H(\phi_0^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{if } A \sqsubseteq \phi_0 \\ \phi_1\ H(\phi_1^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_1 \\ \vdots & \\ \phi_k\ H(\phi_k^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_k \\ H(A \circ \phi_i, \phi_{i_2}\phi_{i_3}\ldots) & \textbf{otherwise.} \end{cases}$$

The rough idea behind the algorithm

When $A(x) \in \phi_j((0, +\infty))$ no matter what x is,
output the digit ϕ_j.
Otherwise, absorb a digit from x to refine our calculation.
Define $A \sqsubseteq \phi_j \Leftrightarrow A([0, +\infty]) \subseteq \phi_j([0, +\infty])$.

$$H(A, \phi_{i_1}\phi_{i_2}\ldots) := \begin{cases} \phi_0 \ H(\phi_0^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{if } A \sqsubseteq \phi_0 \\ \phi_1 \ H(\phi_1^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_1 \\ \quad \vdots & \\ \phi_k \ H(\phi_k^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_k \\ H(A \circ \phi_i, \phi_{i_2}\phi_{i_3}\ldots) & \textbf{otherwise}. \end{cases}$$

The rough idea behind the algorithm

When $A(x) \in \phi_j((0,+\infty))$ no matter what x is, output the digit ϕ_j.
Otherwise, absorb a digit from x to refine our calculation.
Define $A \sqsubseteq \phi_j \Leftrightarrow A([0,+\infty]) \subseteq \phi_j([0,+\infty])$.

$$H(A, \phi_{i_1}\phi_{i_2}\ldots) := \begin{cases} \phi_0 \; H(\phi_0^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{if } A \sqsubseteq \phi_0 \\ \phi_1 \; H(\phi_1^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_1 \\ \vdots & \\ \phi_k \; H(\phi_k^{-1} \circ A, \phi_{i_1}\phi_{i_2}\ldots) & \textbf{else if } A \sqsubseteq \phi_k \\ H(A \circ \phi_i, \phi_{i_2}\phi_{i_3}\ldots) & \textbf{otherwise}. \end{cases}$$