Computability Theory

Barbara Csima

May 7, 2026

EMAIL: csima@uwaterloo.ca
Department of Pure Mathematics
University of Waterloo

Computable Structures

Definition

We say a structure $\mathcal{A}$ is computable if it has domain $\mathbb{N}$ and all functions and relations on $\mathcal{A}$ are computable.

Computable Structures

Definition

We say a structure $\mathcal{A}$ is computable if it has domain $\mathbb{N}$ and all functions and relations on $\mathcal{A}$ are computable.

Example

A linear order is computable if there is a program that for each pair (a, b) computes whether $a < b$ or $b < a$.

Computable Structures

Definition

We say a structure $\mathcal{A}$ is computable if it has domain $\mathbb{N}$ and all functions and relations on $\mathcal{A}$ are computable.

Example

A linear order is computable if there is a program that for each pair (a, b) computes whether $a < b$ or $b < a$.

Example

A directed graph is computable if the edge relation is computable.

Degree Spectra of Structures

Definition

For any structure $\mathcal{A}$, Spec($\mathcal{A}$)= $\{deg(\mathcal{B}) \mid \mathcal{B} \cong \mathcal{A}\}$.

Degree Spectra of Structures

Definition

For any structure $\mathcal{A}$, Spec$(\mathcal{A})$= $\{deg(\mathcal{B}) \mid \mathcal{B} \cong \mathcal{A}\}$.

We are interested in the kinds of degree spectra that natural structures can have.

Degree Spectra of Structures

Definition

For any structure $\mathcal{A}$, Spec$(\mathcal{A})$= $\{deg(\mathcal{B}) \mid \mathcal{B} \cong \mathcal{A}\}$.

We are interested in the kinds of degree spectra that natural structures can have.

Theorem (Julia Knight)

The degree spectra of a non-trivial structure is upward closed in the Turing degrees.

Degree Spectra of Structures

Theorem (Julia Knight)

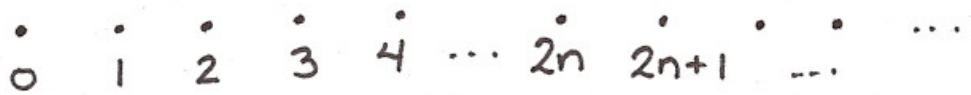
The degree spectra of a non-trivial structure is upward closed in the Turing degrees.

Degree Spectra of Structures

Theorem (Julia Knight)

The degree spectra of a non-trivial structure is upward closed in the Turing degrees.

For example, consider the structure $(\mathbb{N}, <)$:

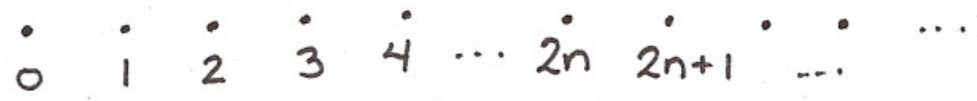


Degree Spectra of Structures

Theorem (Julia Knight)

The degree spectra of a non-trivial structure is upward closed in the Turing degrees.

For example, consider the structure $(\mathbb{N}, <)$:

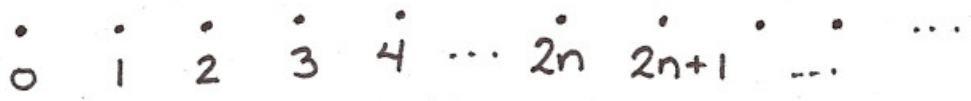


We can code any set A into an isomorphic copy of $(\mathbb{N}, <)$ by simply switching the positions of $2n$ and $2n+1$ iff $n \in A$.

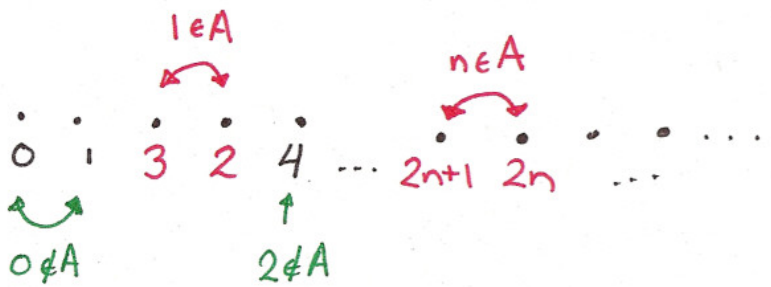
Theorem (Julia Knight)

The degree spectra of a non-trivial structure is upward closed in the Turing degrees.

For example, consider the structure $(\mathbb{N}, <)$:



We can code any set A into an isomorphic copy of $(\mathbb{N}, <)$ by simply switching the positions of $2n$ and $2n+1$ iff $n \in A$.



Degree Spectra of Structures

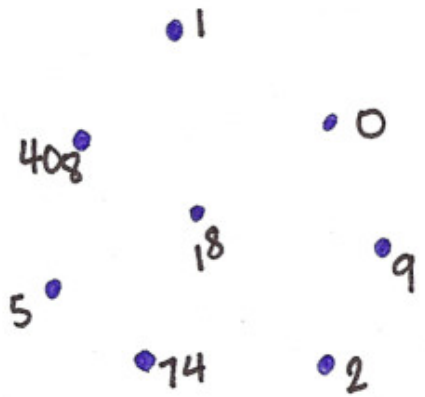
Theorem (Julia Knight)

The degree spectra of a non-trivial structure is upward closed in the Turing degrees.

Theorem (Julia Knight)

The degree spectra of a non-trivial structure is upward closed in the Turing degrees.

However, if the structure is trivial, such as a graph with no edges, there is no place to do coding.



Possible Degree Spectra

Theorem (Knight)

If C is an at most countable set of incomparable Turing degrees, then there is no structure whose degree spectrum is exactly those degrees computing the members of C.

Possible Degree Spectra

Theorem (Knight)

If C is an at most countable set of incomparable Turing degrees, then there is no structure whose degree spectrum is exactly those degrees computing the members of C.

Theorem (Slaman, Wehner)

There is a structure whose degree spectrum is exactly the non-computable degrees.

Theorem (Downey-Jockusch)

If a boolean algebra $\mathcal{B}$ has a low copy, then it has a computable copy.

Degree Spectra of Boolean Algebras

Theorem (Downey-Jockusch)

If a boolean algebra $\mathcal{B}$ has a low copy, then it has a computable copy.

Corollary

There is no boolean algebra who's degree spectrum is exactly the non-computable degrees.

Computable Categoricity

Definition

A computable structure $\mathcal{A}$ is computably categorical if for all computable $\mathcal{B} \cong \mathcal{A}$ there exists a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Computable Categoricity

Definition

A computable structure $\mathcal{A}$ is computably categorical if for all computable $\mathcal{B} \cong \mathcal{A}$ there exists a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.

Computable Categoricity

Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.

Computable Categoricity

Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.

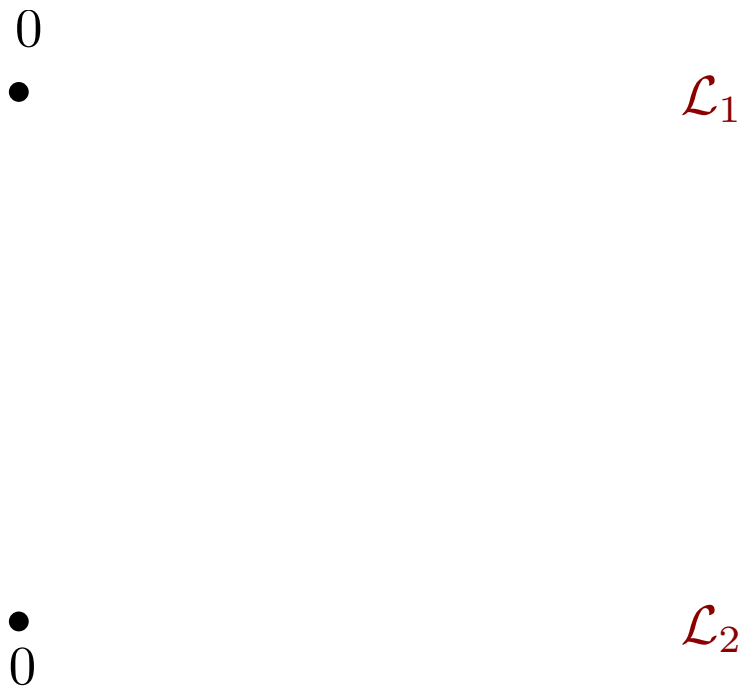
$\mathcal{L}_1$

$\mathcal{L}_2$

Computable Categoricity

Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



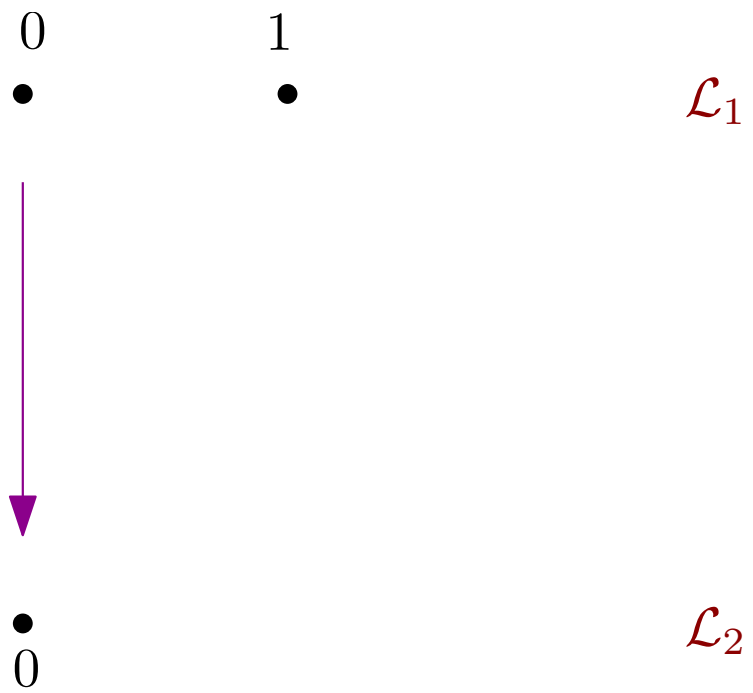
Computable Categoricity

Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.

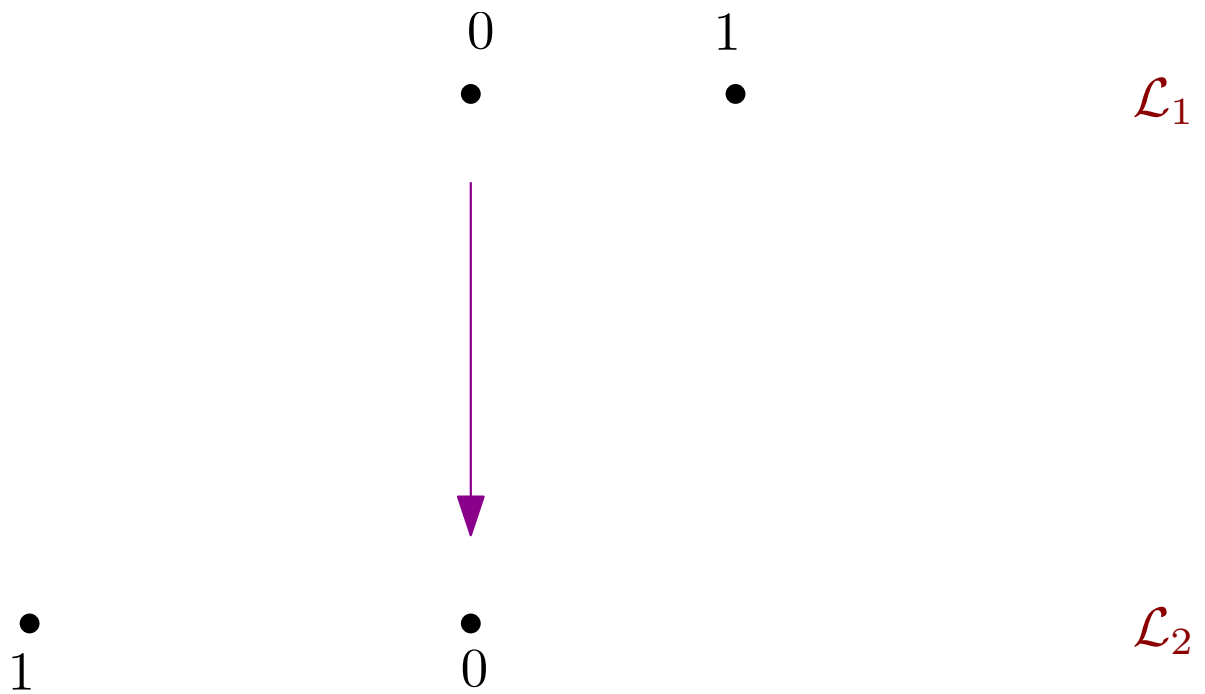
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



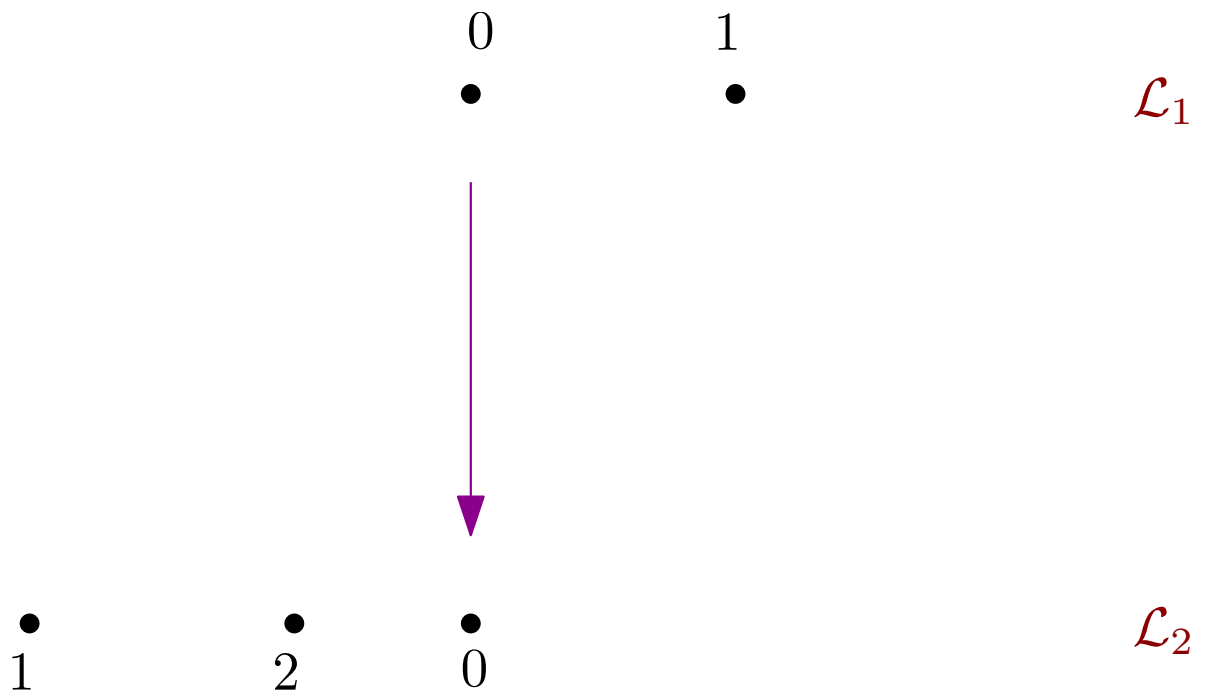
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



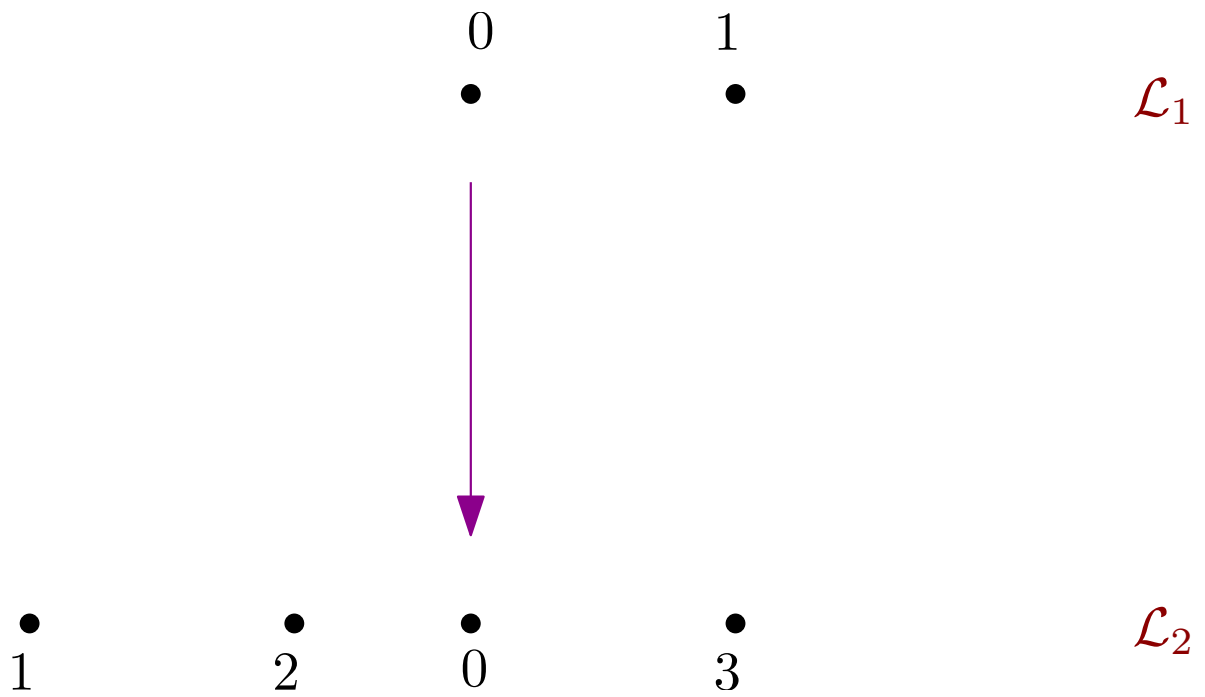
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



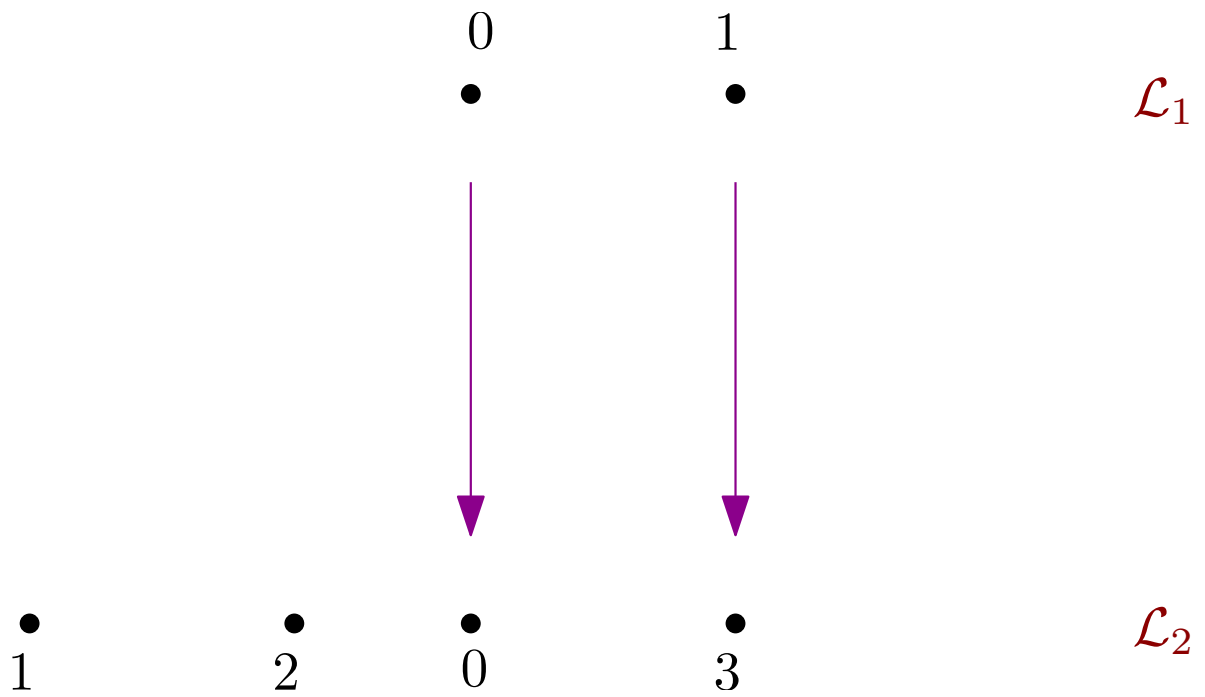
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



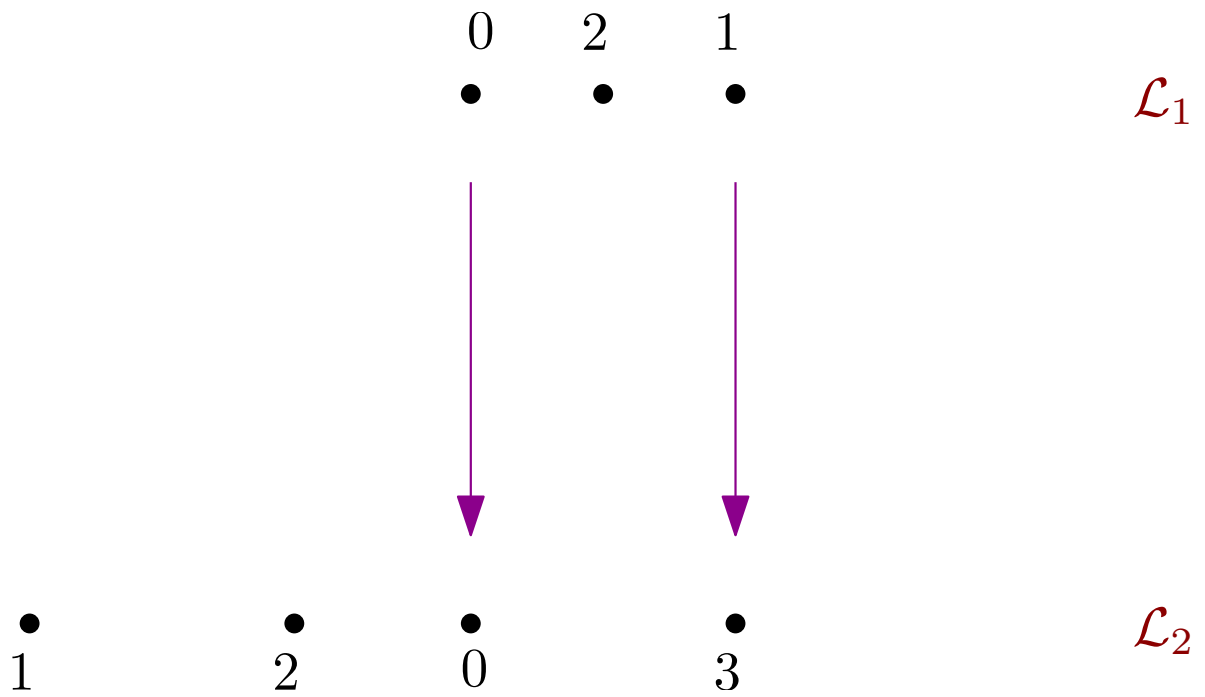
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



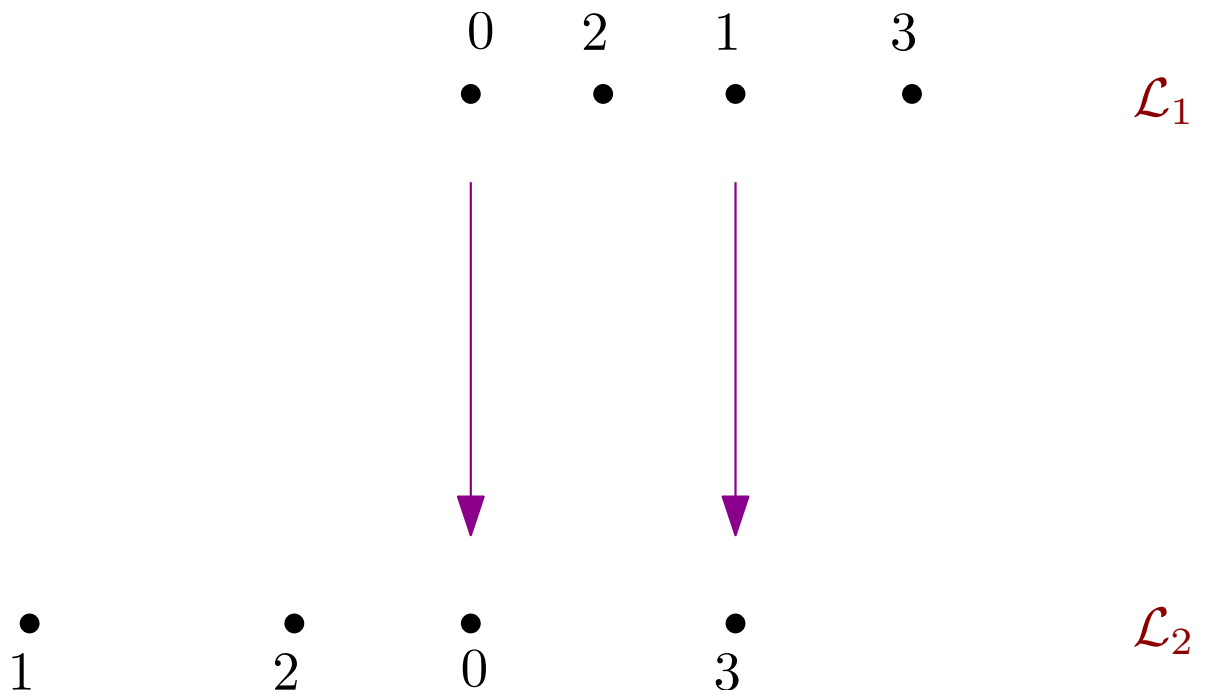
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



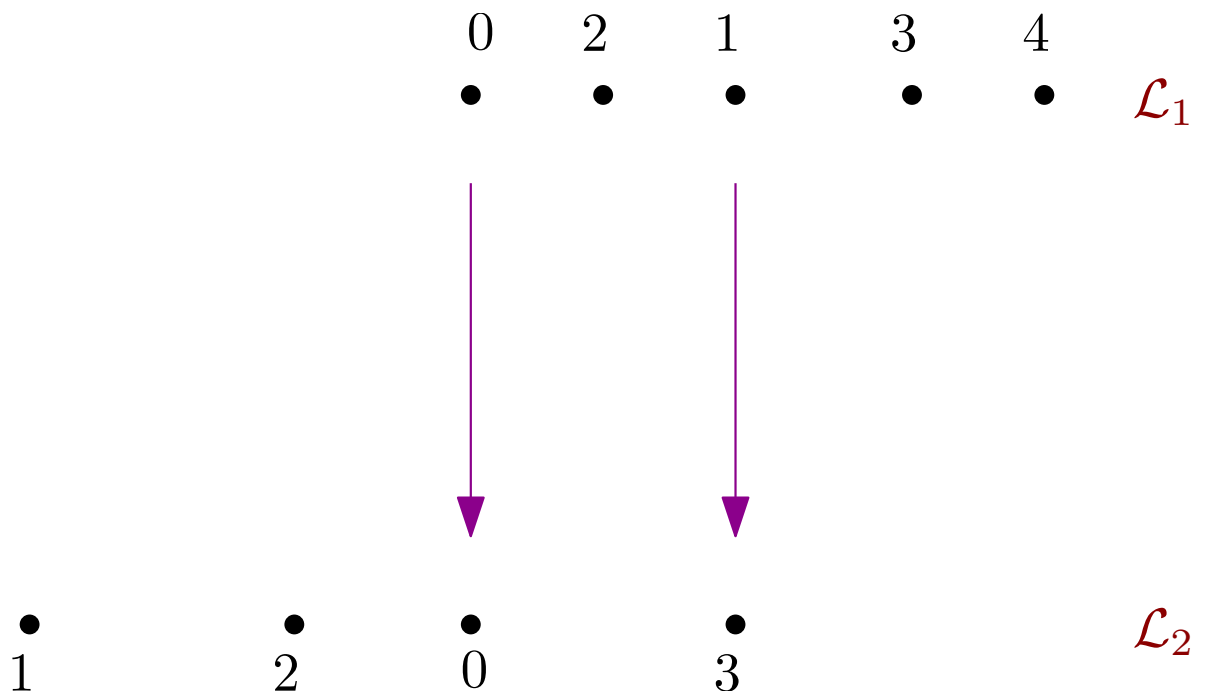
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



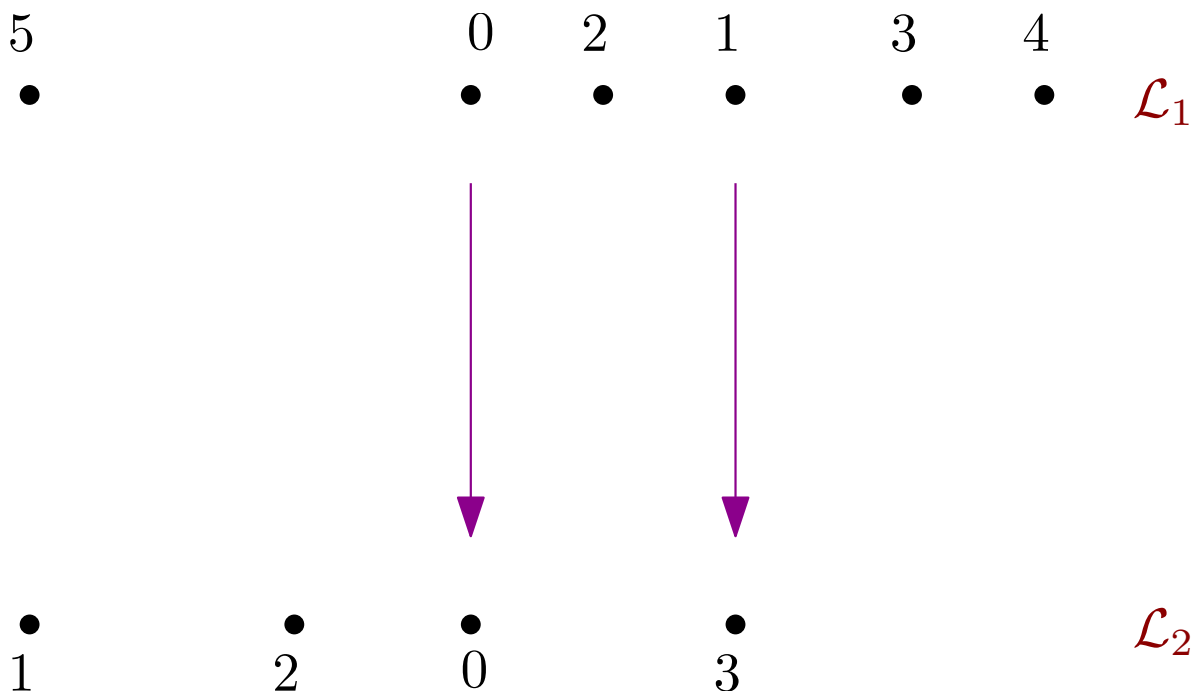
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



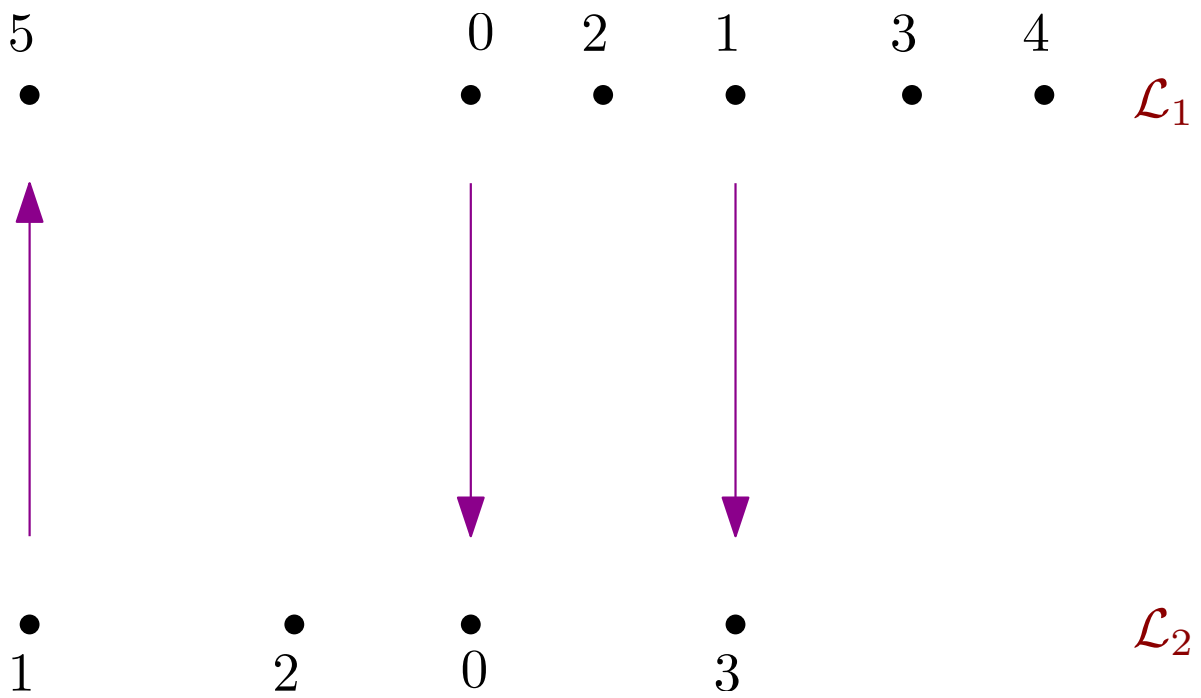
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



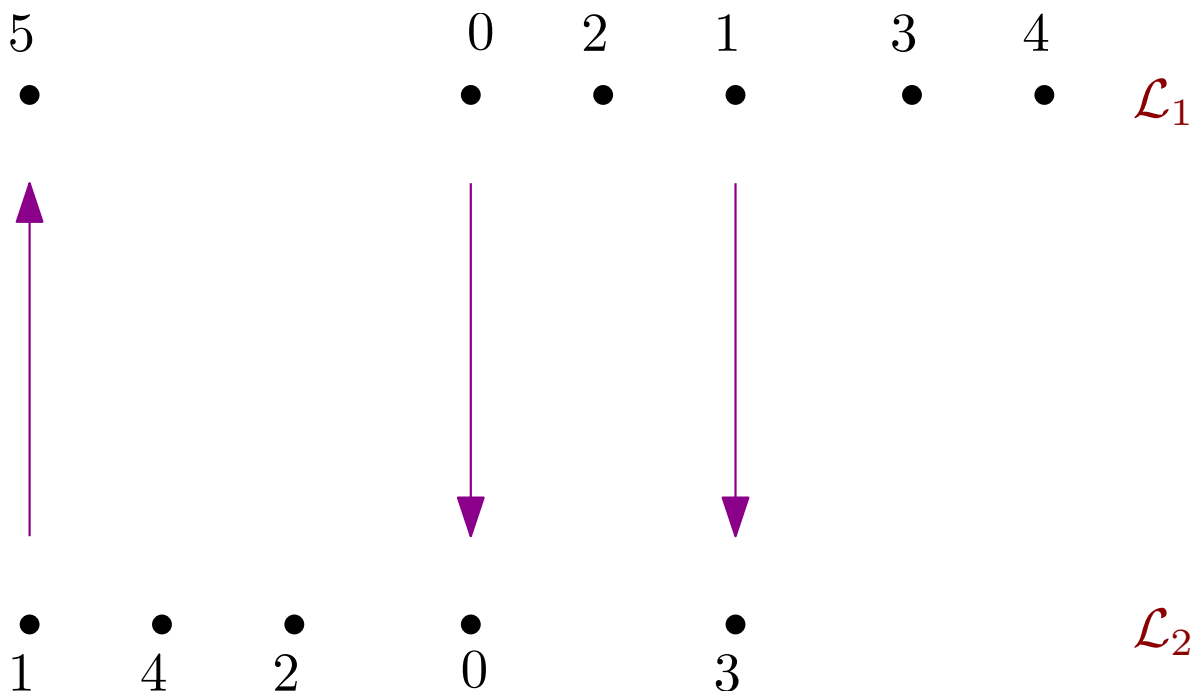
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



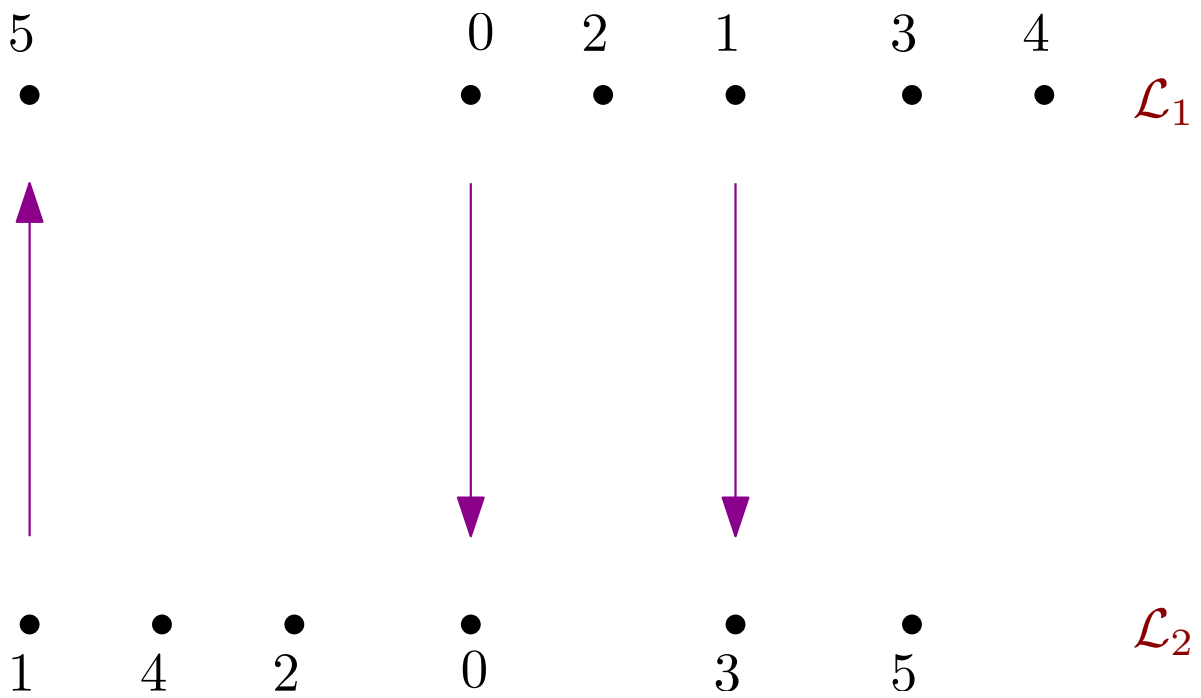
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



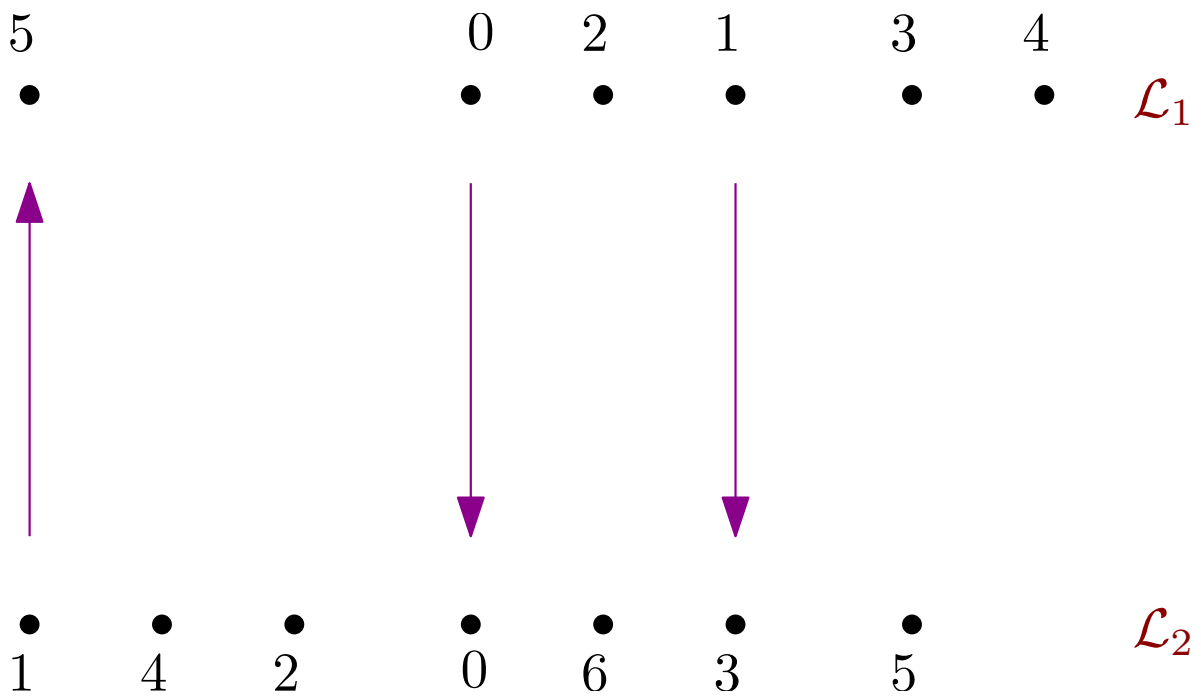
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



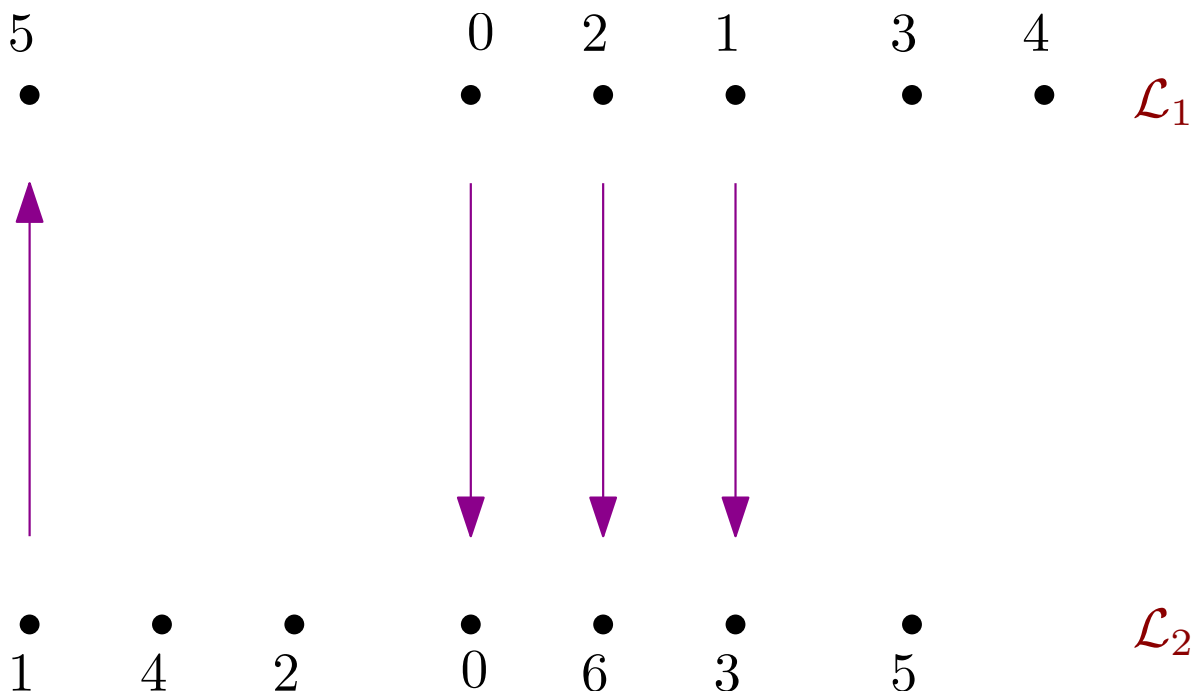
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



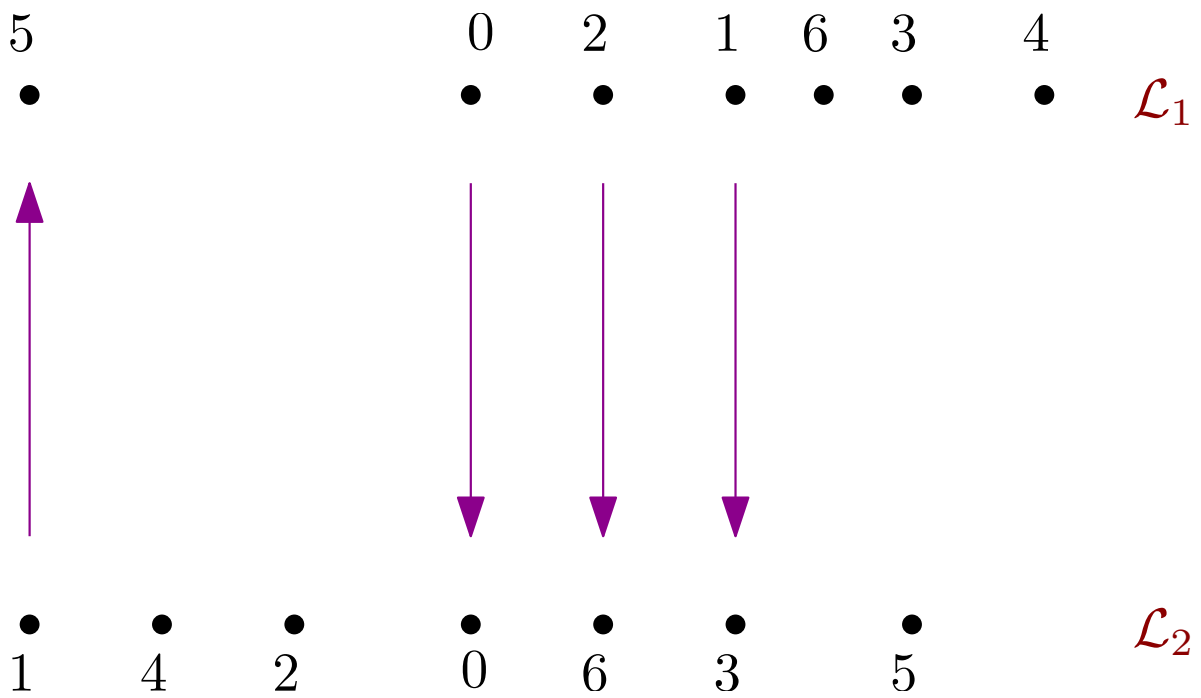
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



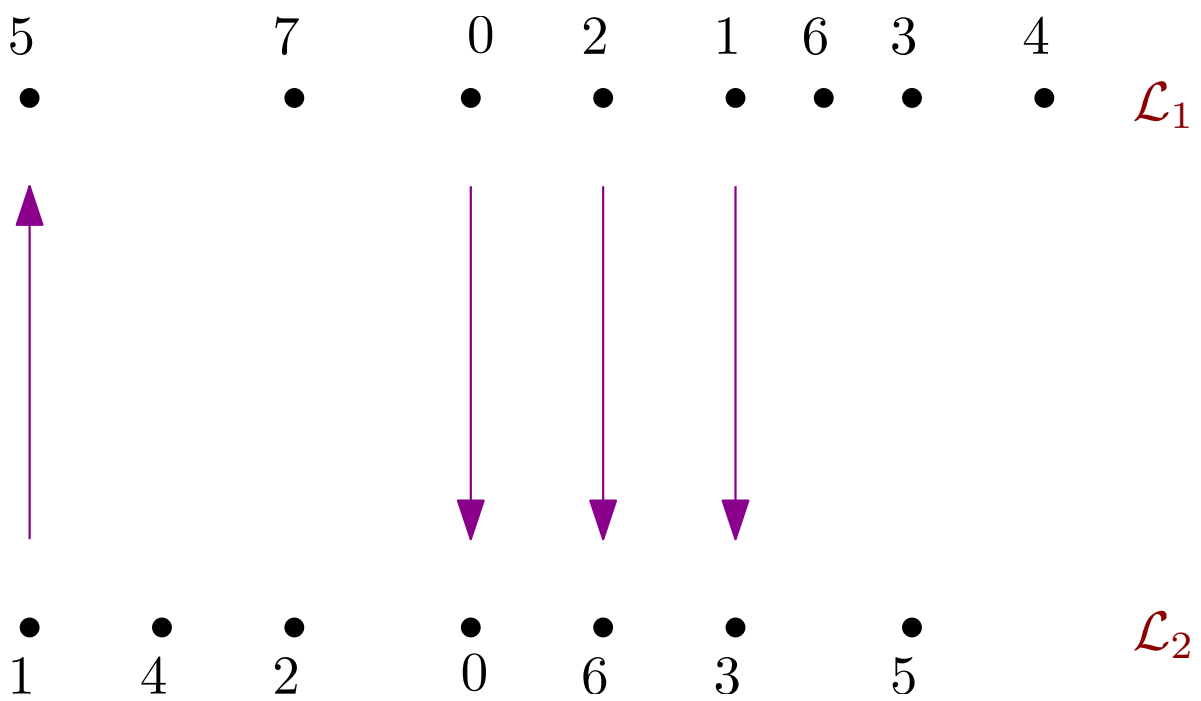
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



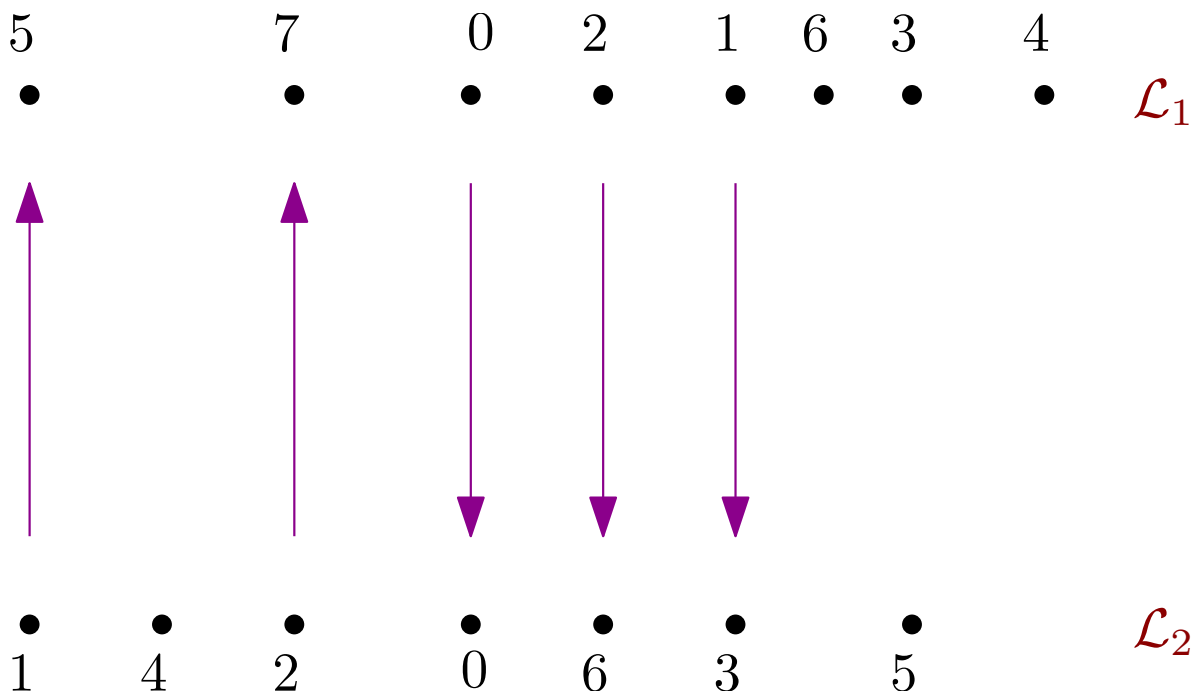
Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



Example

Any two computable dense linear orders without endpoints are computably isomorphic. Thus, any computable dense linear order without endpoints is computably categorical.



Example

Consider the structure $(\mathbb{N}, <)$, the natural numbers with the usual $<$ order.

Example

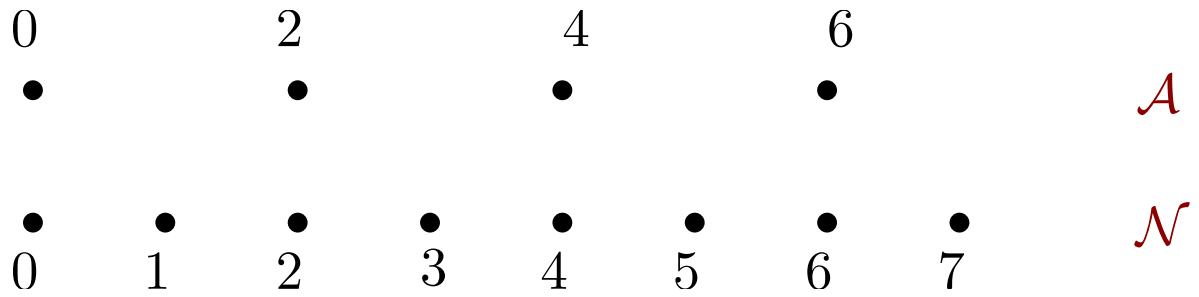
Consider the structure $(\mathbb{N}, <)$, the natural numbers with the usual $<$ order.

Let $\{K_s\}_{s\in\omega}$ be a nice enumeration of $\varnothing'$ (the halting set), and consider the order $\mathcal{A}$ where the even numbers have their usual order and $2n <_{\mathcal{A}} 2s+1 <_{\mathcal{A}} 2n+2$ iff $n \in K_s - K_{s-1}$.

Example

Consider the structure $(\mathbb{N}, <)$, the natural numbers with the usual $<$ order.

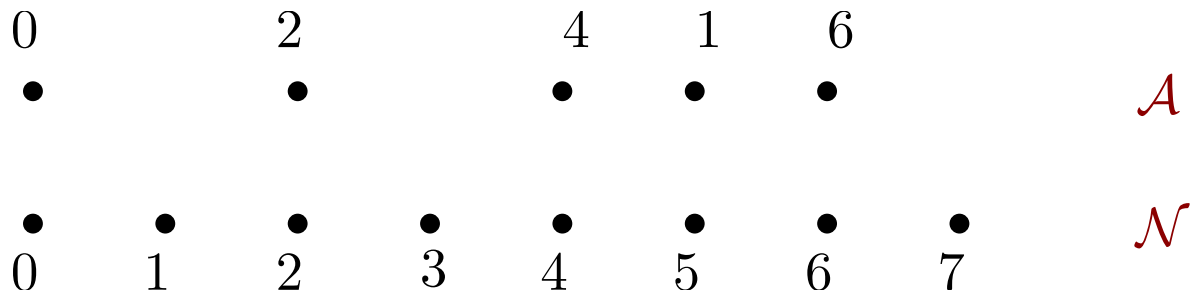
Let $\{K_s\}_{s\in\omega}$ be a nice enumeration of $\varnothing'$ (the halting set), and consider the order $\mathcal{A}$ where the even numbers have their usual order and $2n <_{\mathcal{A}} 2s+1 <_{\mathcal{A}} 2n+2$ iff $n \in K_s - K_{s-1}$.



Example

Consider the structure $(\mathbb{N}, <)$, the natural numbers with the usual $<$ order.

Let $\{K_s\}_{s\in\omega}$ be a nice enumeration of $\varnothing'$ (the halting set), and consider the order $\mathcal{A}$ where the even numbers have their usual order and $2n <_{\mathcal{A}} 2s+1 <_{\mathcal{A}} 2n+2$ iff $n \in K_s - K_{s-1}$.

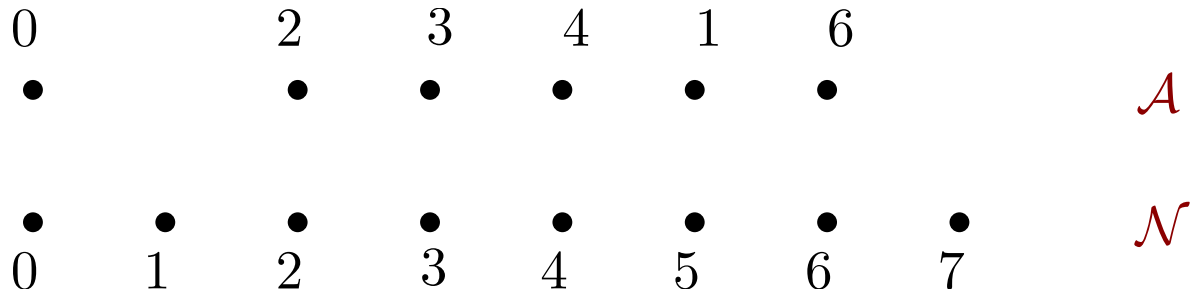


$2 \in K_0$

Example

Consider the structure $(\mathbb{N}, <)$, the natural numbers with the usual $<$ order.

Let $\{K_s\}_{s\in\omega}$ be a nice enumeration of $\varnothing'$ (the halting set), and consider the order $\mathcal{A}$ where the even numbers have their usual order and $2n <_{\mathcal{A}} 2s+1 <_{\mathcal{A}} 2n+2$ iff $n \in K_s - K_{s-1}$.

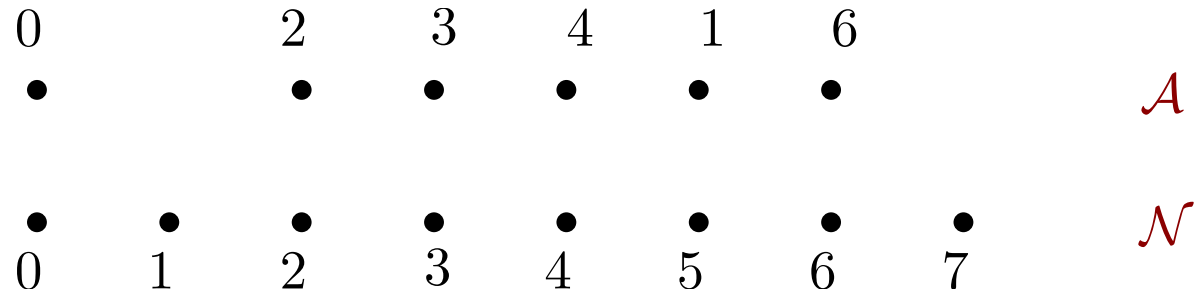


$1 \in K_1$

Example

Consider the structure $(\mathbb{N}, <)$, the natural numbers with the usual $<$ order.

Let $\{K_s\}_{s\in\omega}$ be a nice enumeration of $\varnothing'$ (the halting set), and consider the order $\mathcal{A}$ where the even numbers have their usual order and $2n <_{\mathcal{A}} 2s+1 <_{\mathcal{A}} 2n+2$ iff $n \in K_s - K_{s-1}$.

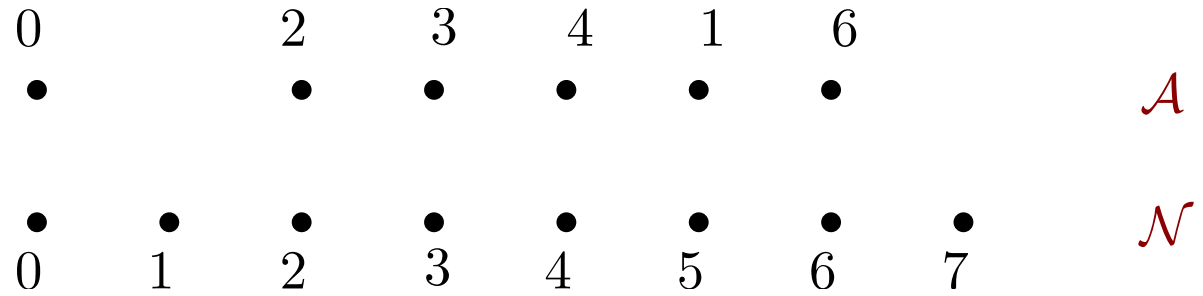


Note that any isomorphism $f : \mathcal{A} \to \mathcal{N}$ computes $\varnothing'$.

Example

Consider the structure $(\mathbb{N}, <)$, the natural numbers with the usual $<$ order.

Let $\{K_s\}_{s\in\omega}$ be a nice enumeration of $\varnothing'$ (the halting set), and consider the order $\mathcal{A}$ where the even numbers have their usual order and $2n <_{\mathcal{A}} 2s+1 <_{\mathcal{A}} 2n+2$ iff $n \in K_s - K_{s-1}$.



Note that any isomorphism $f : \mathcal{A} \to \mathcal{N}$ computes $\varnothing'$.
Conversely, between any two computable copies of $(\mathbb{N}, <)$, we can construct an isomorphism if we can answer existential questions about the order, hence there is a $\varnothing'$-computable isomorphism.

d- Computable Categoricity

Definition

A computable structure $\mathcal{A}$ is **d**- computably categorical if for all computable $\mathcal{B} \cong \mathcal{A}$ there exists a **d**- computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

d- Computable Categoricity

Definition

A computable structure $\mathcal{A}$ is **d**- computably categorical if for all computable $\mathcal{B} \cong \mathcal{A}$ there exists a **d**- computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Example

$(\mathbb{N}, <)$ is $0'$-**computably categorical.**

Degrees of Categoricity

Definition (Fokina, Kalimullin, Miller)

We say $\mathbf{d}$ is a degree of categoricity if there exists a computable structure $\mathcal{A}$ such that

- There exists a $\mathbf{d}$-computable isomorphism between any two computable copies of $\mathcal{A}$.
- Any other degree with this property computes $\mathbf{d}$.

Degrees of Categoricity

Definition (Fokina, Kalimullin, Miller)

We say $\mathbf{d}$ is a degree of categoricity if there exists a computable structure $\mathcal{A}$ such that

- There exists a $\mathbf{d}$-computable isomorphism between any two computable copies of $\mathcal{A}$.
- Any other degree with this property computes $\mathbf{d}$.

Example

$\mathbf{0}'$ is a degree of categoricity, as witnessed by $(\mathbb{N}, <)$.

Approximations of Δ^0_2 sets

The Limit Lemma tells us that a set A satisfies $A \leq_T \varnothing'$ if and only if there exists a computable function $f(x,s)$ such that $A(x) = \lim_s f(x,s)$.

Approximations of Δ^0_2 sets

The Limit Lemma tells us that a set A satisfies $A \leq_T \varnothing'$ if and only if there exists a computable function $f(x,s)$ such that $A(x) = \lim_s f(x,s)$.

Definition

A set A is said to be n-c.e. if there is a computable function $f(x,s)$ such that $f(x,0) = 0$, $A(x) = \lim_s f(x,s)$, and $|\{s \mid f(x,s) \neq f(x,s+1)\}| \leq n$.

Degrees of Categoricity

Theorem (Fokina, Kalimullin, Miller, 2010)

If $\mathbf{d}$ *is 2-c.e. in and above* $\mathbf{0^{(n)}}$*, then* $\mathbf{d}$ *is a degree of categoricity.*

$\mathbf{0}^{(\omega)}$ *is a degree of categoricity.*

C.e. degrees are degrees of categoricity

Let A be a c.e. set.

C.e. degrees are degrees of categoricity

Let A be a c.e. set. We build a graph with degree of categoricity $\deg(A)$.

C.e. degrees are degrees of categoricity

Let A be a c.e. set. We build a graph with degree of categoricity $\deg(A)$. The graph consists of an omega chain, and to the nth node, we link the following graph, depending on whether $n \in A$.

C.e. degrees are degrees of categoricity

Let A be a c.e. set. We build a graph with degree of categoricity $\deg(A)$. The graph consists of an omega chain, and to the nth node, we link the following graph, depending on whether $n \in A$.

If $n \notin A$:

C.e. degrees are degrees of categoricity

Let A be a c.e. set. We build a graph with degree of categoricity $\deg(A)$. The graph consists of an omega chain, and to the nth node, we link the following graph, depending on whether $n \in A$.

If $n \in A$:

2-c.e. are degrees of categoricity

Let D be a 2-c.e set.

2-c.e. are degrees of categoricity

Let D be a 2-c.e set. We proceed as before, except that for each n we now use two coding locations, and the resulting graph will depend on a 2-c.e. enumeration of D.

2-c.e. are degrees of categoricity

Let D be a 2-c.e set. We proceed as before, except that for each n we now use two coding locations, and the resulting graph will depend on a 2-c.e. enumeration of D.
While $n \notin D$, in the first coding location:

In the second coding location:

(empty)

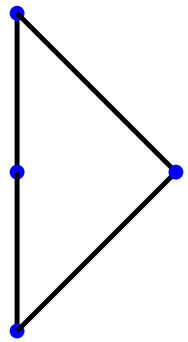
2-c.e. are degrees of categoricity

Let D be a 2-c.e set. We proceed as before, except that for each n we now use two coding locations, and the resulting graph will depend on a 2-c.e. enumeration of D.
If n enters D, in the first coding location:

In the second coding location:

2-c.e. are degrees of categoricity

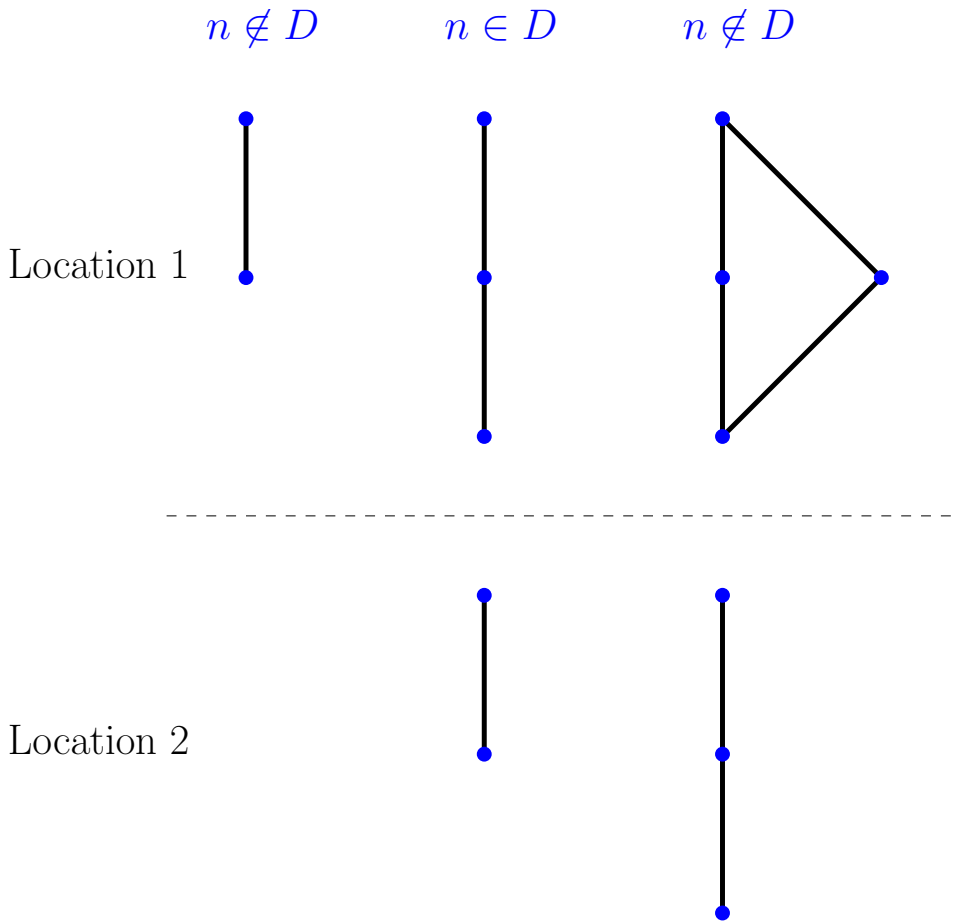
Let D be a 2-c.e set. We proceed as before, except that for each n we now use two coding locations, and the resulting graph will depend on a 2-c.e. enumeration of D.
If n exits D, in the first coding location:



In the second coding location:

2-c.e. are degrees of categoricity

Let D be a 2-c.e set. We proceed as before, except that for each n we now use two coding locations, and the resulting graph will depend on a 2-c.e. enumeration of D.



Δ^0_2 degrees are degrees of categoricity

Theorem (C, Ng)

Every Δ^0_2 degree is a strong degree of categoricity.

Δ^0_2 degrees are degrees of categoricity

Theorem (C, Ng)

Every Δ^0_2 degree is a strong degree of categoricity.

What was the difficulty compared with 2-c.e.?

Δ^0_2 degrees are degrees of categoricity

Theorem (C, Ng)

Every Δ^0_2 degree is a strong degree of categoricity.

What was the difficulty compared with 2-c.e.?

In the 2-c.e. coding, one has two distinct nodes that look different. If the first c.e. event happens, we grow each distinct node to still be different, but change the unique isomorphism. If the second c.e. event happens, we make them look the same.

Δ^0_2 degrees are degrees of categoricity

Theorem (C, Ng)

Every Δ^0_2 degree is a strong degree of categoricity.

What was the difficulty compared with 2-c.e.?

In the 2-c.e. coding, one has two distinct nodes that look different. If the first c.e. event happens, we grow each distinct node to still be different, but change the unique isomorphism. If the second c.e. event happens, we make them look the same.

If we try to introduce more coding locations in the same way, we end up losing the D-categoricity of the structure.

Δ^0_2 degrees are degrees of categoricity

How did we overcome the difficulty?

How did we overcome the difficulty?

To code whether a particular $n \in D$, we introduce new coding locations every time there is a change in the approximation. By consulting an isomorphism on a given coding location between our distinct copies, we learn either that the approximation to D on n is correct at the stage, or that there is a further coding location introduced. This is basically like in the naive strategy.

How did we overcome the difficulty?

To code whether a particular $n \in D$, we introduce new coding locations every time there is a change in the approximation. By consulting an isomorphism on a given coding location between our distinct copies, we learn either that the approximation to D on n is correct at the stage, or that there is a further coding location introduced. This is basically like in the naive strategy.

In order to make sure our structure is D-categorical, we keep track of all possible computable copies of our structure and build a D-computable isomorphism between them. This uses an infinite injury priority argument.

Δ^0_2 degrees are degrees of categoricity

The main trick in the basic module, is that when we "de-homogonize" a coding location, we have to be able to correctly recover a D-computable isomorphism. So we make infinite chains of nodes that look distinct stage-by-stage to control recovery, but such that they are isomorphic in the limit.

Δ^0_2 degrees are degrees of categoricity

The main trick in the basic module, is that when we "de-homogonize" a coding location, we have to be able to correctly recover a D-computable isomorphism. So we make infinite chains of nodes that look distinct stage-by-stage to control recovery, but such that they are isomorphic in the limit.

$$\cdots \to \mathbb{B}^2_\eta(\mu_L) \to \mathbb{B}_\eta(\mu_L) \to \mu_L \to \mathbb{F}_\eta(\mu_L) \to \mathbb{F}^2_\eta(\mu_L) \to \cdots$$

$$\cdots \to \mathbb{B}^2_\eta(\mu_R) \to \mathbb{B}_\eta(\mu_R) \to \mu_R \to \mathbb{F}_\eta(\mu_R) \to \mathbb{F}^2_\eta(\mu_R) \to \cdots$$