It's All Relative, Except When It's Not: Categoricity Versus Relative Categoricity

Asher M. Kach

Joint Work with Rod Downey, Steffen Lempp, Andy Lewis, Antonio Montalbán, and Dan Turetsky

Cornell University
5 September 2012

Outline

Computable and Relative Computable Categoricity...

Definition

A computable structure $\mathcal{M}$ is *computably categorical* if between any two computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is a computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.

Definition

A computable structure $\mathcal{M}$ is *relatively computably categorical* if between any two arbitrary presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is a $(\mathcal{A} \oplus \mathcal{B})$-computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.

Computable and Relative Computable Categoricity...

Definition

A computable structure $\mathcal{M}$ is *computably categorical* if between any two computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is a computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.

Definition

A computable structure $\mathcal{M}$ is *relatively computably categorical* if between any two arbitrary presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is a $(\mathcal{A} \oplus \mathcal{B})$-computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.

Question

How and why are these notions the same? Different?

Question

Are there structural properties that are common to exactly these structures?

Relativizing The Definitions...

Definition

A computable structure $\mathcal{M}$ is Δ^0_α*-categorical* if between any two computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is a Δ^0_α-computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.

Definition

A computable structure $\mathcal{M}$ is *relatively* Δ^0_α*-categorical* if between any two arbitrary presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is a $(\Delta^0_\alpha(\mathcal{A}) \oplus \Delta^0_\alpha(\mathcal{B}))$-computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.

A Syntactic Characterization...

A Syntactic Characterization...

Theorem (Goncharov 1975; Ash 1987)

For a computable structure $\mathcal{M}$, the following are equivalent:

- *The structure $\mathcal{M}$ is relatively Δ^0_α-categorical.*
- *The orbits are effectively isolated by Σ^c_α-formulas: There is a c.e. family Φ of Σ^c_α-formulas (over some fixed parameter $\overline{c}$) such that each $\overline{a} \in M$ satisfies some $\varphi \in \Phi$, and if $\overline{a}, \overline{b} \in M$ both satisfy the same $\varphi \in \Phi$ then they are automorphic.*
- *The Σ^c_α-types are effectively isolated by Σ^c_α-formulas: There is a c.e. family Φ of Σ^c_α-formulas (over some fixed parameter $\overline{c}$) such that each $\overline{a} \in M$ satisfies some $\varphi \in \Phi$, and if $\overline{a}, \overline{b} \in M$ both satisfy the same $\varphi \in \Phi$ then their Σ^c_α-types coincide.*

Uniform Computable Categoricity...

Definition (Kudinov 1996)

A computable structure $\mathcal{M}$ is *uniformly computably categorical* if there is a functional Φ such that

$$x \mapsto \Phi(AtDiag(\mathcal{M}_i), AtDiag(\mathcal{M}_j); x)$$

is a computable isomorphism between $\mathcal{M}_i$ and $\mathcal{M}_j$ whenever $\mathcal{M}_i$ and $\mathcal{M}_j$ are computable presentations of $\mathcal{M}$.

A computable structure $\mathcal{M}$ is *uniformly computably categorical with parameters* if there is a parameter $\overline{c} \in M$ such that $(\mathcal{M}; \overline{c})$ is uniformly computably categorical.

The Connection...

Theorem (Ash, Knight, and Slaman 1993; Downey, Hirschfeldt, and Khoussainov 2003)

A structure $\mathcal{M}$ is relatively computably categorical (with parameters) if and only if it is uniformly computably categorical (with parameters).

The Connection...

Theorem (Ash, Knight, and Slaman 1993; Downey, Hirschfeldt, and Khoussainov 2003)

A structure $\mathcal{M}$ is relatively computably categorical (with parameters) if and only if it is uniformly computably categorical (with parameters).

Proof ($\Longrightarrow$).

Fix Φ witnessing relative computable categoricity. Given computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$, define $\pi : \mathcal{A} \cong \mathcal{B}$ via a back-and-forth construction.

For the forth direction, to define $\pi(a)$, search for $\varphi \in \Phi$ such that $\mathcal{A} \models \varphi(x_0, x_1, \ldots, x_i, a)$, where $x_0, x_1, \ldots, x_i$ are the elements already in the domain of π.

Search for $b \in B$ such that $\mathcal{B} \models \varphi(\pi(x_0), \pi(x_1), \ldots, \pi(x_i), b)$. Define $\pi(a) = b$. $\square$

The Connection...

Theorem (Ash, Knight, and Slaman 1993; Downey, Hirschfeldt, and Khoussainov 2003)

A structure $\mathcal{M}$ is relatively computably categorical (with parameters) if and only if it is uniformly computably categorical (with parameters).

Proof ($\Longleftarrow$).

Fix computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$. Build family Φ.

Given $\overline{a} \in A$, let s be least so that

$$\overline{a} \subseteq \mathcal{A} \upharpoonright s \quad \text{and} \quad \Psi(\mathrm{AtDiag}(\mathcal{A}) \upharpoonright s, \mathrm{AtDiag}(\mathcal{B}) \upharpoonright s; x).$$

Let $\delta(\overline{a}, \overline{b})$ be the atomic diagram given by $\mathcal{A} \upharpoonright s$.

Define $\varphi_{\overline{a}}(\overline{x})$ to be the formula $\varphi_{\overline{a}}(\overline{x}) := (\exists y)\,[\varphi(\overline{x}, \overline{y})]$.

$\square$

Outline

Index Set Complexity...

Corollary (Downey, Kach, Lempp, and Turetsky)

The index set $\{e : \mathcal{M}_e$ *is relatively computably categorical*$\}$ *of the relatively computably categorical structures is* Σ^0_3*-complete.*

Index Set Complexity...

Corollary (Downey, Kach, Lempp, and Turetsky)

The index set $\{e : \mathcal{M}_e$ is relatively computably categorical$\}$ of the relatively computably categorical structures is Σ^0_3-complete.

Proof.

For Σ^0_3: Analyze Goncharov 1975: By the third condition, relatively computably categorical if and only if

$$(\exists\Phi)\Big[(\forall\overline{a})(\exists\varphi\in\Phi)\Big[\mathcal{M}_e\models\varphi(\overline{a})\Big] \text{ and } (\forall\overline{a})(\forall\overline{b})(\forall\psi\in\Sigma^0_1)(\forall\theta\in\Sigma^0_1)$$
$$\Big[(\mathcal{M}_e\models\psi(\overline{a})\wedge\psi(\overline{b}) \text{ and } \psi\in\Phi)\rightarrow\mathcal{M}_e\models\theta(\overline{a})\wedge\theta(\overline{b})\Big]\Big].$$

Index Set Complexity...

Corollary (Downey, Kach, Lempp, and Turetsky)

The index set $\{e : \mathcal{M}_e$ is relatively computably categorical$\}$ of the relatively computably categorical structures is Σ^0_3-complete.

Proof.

For Σ^0_3: Analyze Goncharov 1975: By the third condition, relatively computably categorical if and only if

$$(\exists\Phi)\Big[(\forall\overline{a})(\exists\varphi\in\Phi)\Big[\mathcal{M}_e\models\varphi(\overline{a})\Big] \text{ and } (\forall\overline{a})(\forall\overline{b})(\forall\psi\in\Sigma^0_1)(\forall\theta\in\Sigma^0_1)$$
$$\Big[(\mathcal{M}_e\models\psi(\overline{a})\wedge\psi(\overline{b}) \text{ and } \psi\in\Phi)\rightarrow\mathcal{M}_e\models\theta(\overline{a})\wedge\theta(\overline{b})\Big]\Big].$$

For Hardness: Exploit, for example, Downey and Montalbán 2008: A $\mathbb{Z}$-vector space is relatively computably categorical if and only if it is finite dimensional. □

Index Set Complexity...

Theorem (Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky)

The index set $\{e : \mathcal{M}_e$ *is computably categorical*$\}$ *of the computably categorical structures is* Π^1_1*-complete.*

Index Set Complexity...

Theorem (Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky)

The index set $\{e : \mathcal{M}_e$ is computably categorical$\}$ of the computably categorical structures is Π^1_1-complete.

Proof.

For Π^1_1: Note that $\mathcal{M}_e$ is computably categorical if and only if

$$(\forall i)\,[(\exists f)\,[f : \mathcal{M}_e \cong \mathcal{M}_i] \implies (\exists k)\,[\Phi_k : \mathcal{M}_e \cong \mathcal{M}_i]]\,.$$

For Hardness: Not enough space in the margins. □

Index Set Complexity...

Theorem (Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky)

The index set $\{e : \mathcal{M}_e$ is computably categorical$\}$ of the computably categorical structures is Π^1_1-complete.

Proof.

For Π^1_1: Note that $\mathcal{M}_e$ is computably categorical if and only if

$$(\forall i)\,[(\exists f)\,[f : \mathcal{M}_e \cong \mathcal{M}_i] \implies (\exists k)\,[\Phi_k : \mathcal{M}_e \cong \mathcal{M}_i]]\,.$$

For Hardness: Not enough space in the margins. □

Remark

In a strong sense, this says there is no simple structural property whose presence or absence is equivalent to computable categoricity.

Index Set Complexity...

Theorem (Downey, Kach, Lempp, Melnikov, and Turetsky)

There is a relatively computably categorical group $\mathcal{G}$ whose index set $\{e : \mathcal{G} \cong \mathcal{G}_e\}$ is Σ^0_3-complete.

Proof.

Index Set Complexity...

Theorem (Downey, Kach, Lempp, Melnikov, and Turetsky)

There is a relatively computably categorical group $\mathcal{G}$ whose index set $\{e : \mathcal{G} \cong \mathcal{G}_e\}$ is Σ^0_3-complete.

Proof.

For Σ^0_3: Fix a (relatively) computably categorical structure $\mathcal{M}$. Then $i \in \{e : \mathcal{M} \cong \mathcal{M}_e\}$ if and only if there is a computable isomorphism

- total (Π^0_2),
- injective (Π^0_1),
- surjective (Π^0_2), and
- atomic diagram preserving (Π^0_1) map

between $\mathcal{M}$ and $\mathcal{M}_i$.

$\square$

Index Set Complexity...

Theorem (Downey, Kach, Lempp, Melnikov, and Turetsky)

There is a relatively computably categorical group $\mathcal{G}$ whose index set $\{e : \mathcal{G} \cong \mathcal{G}_e\}$ is Σ^0_3-complete.

Proof.

For Hardness: Let $\mathcal{G}$ be the subgroup of $(\mathbb{Q}; +)$ generated by the set $\left\{\frac{1}{p} : p \text{ a prime}\right\}$.

Then $\mathcal{G}$ is relatively computably categorical because any isomorphism is dictated by the image of any element.

Given a Σ^0_3 set S with $n \in S$ if and only if $(\exists u)(\exists^\infty v)\,[R(n, u, v)]$, build $\{\mathcal{G}_n\}_{n\in\mathbb{N}}$ so that $\mathcal{G} \cong \mathcal{G}_n$ if and only if $n \in S$. For each n, maintain markers on the prime numbers. When $R(n, u, v)$, kick the uth marker, enumerating $1/p$ into $\mathcal{G}_n$, where p was the prime covered by the uth marker. $\square$

Outline

The 2-Decidable Hypothesis...

Theorem (Goncharov 1975)

Let $\mathcal{M}$ be 2-decidable. If $\mathcal{M}$ is computably categorical, then $\mathcal{M}$ is relatively computably categorical.

The 2-Decidable Hypothesis...

Theorem (Goncharov 1975)

Let $\mathcal{M}$ be 2-decidable. If $\mathcal{M}$ is computably categorical, then $\mathcal{M}$ is relatively computably categorical.

Theorem (Kudinov 1996)

This is optimal: There is a 1-decidable structure $\mathcal{M}$ that is computably categorical but not relatively computably categorical.

The 1-Decidable Hypothesis...

Theorem (Downey, Kach, Lempp, and Turetsky)

Let $\mathcal{M}$ be 1-decidable. If $\mathcal{M}$ is computably categorical, then $\mathcal{M}$ is relatively Δ^0_2 categorical.

The 1-Decidable Hypothesis...

Theorem (Downey, Kach, Lempp, and Turetsky)

Let $\mathcal{M}$ be 1-decidable. If $\mathcal{M}$ is computably categorical, then $\mathcal{M}$ is relatively Δ^0_2 categorical.

Lemma

There is a tuple $\overline{p}$ such that distinct Σ_1-types over $\overline{p}$ are incomparable under inclusion and $\Sigma_1 - tp_{\overline{p}}(\overline{a}) = \Sigma_1 - tp_{\overline{p}}(\overline{a}')$ implies $\Sigma^c_2 - tp_{\overline{p}}(\overline{a}) = \Sigma^c_2 - tp_{\overline{p}}(\overline{a}')$.

The 1-Decidable Hypothesis...

Theorem (Downey, Kach, Lempp, and Turetsky)

Let $\mathcal{M}$ be 1-decidable. If $\mathcal{M}$ is computably categorical, then $\mathcal{M}$ is relatively Δ^0_2 categorical.

Lemma

There is a tuple $\overline{p}$ such that distinct Σ_1-types over $\overline{p}$ are incomparable under inclusion and $\Sigma_1 - tp_{\overline{p}}(\overline{a}) = \Sigma_1 - tp_{\overline{p}}(\overline{a}')$ implies $\Sigma^c_2 - tp_{\overline{p}}(\overline{a}) = \Sigma^c_2 - tp_{\overline{p}}(\overline{a}')$.

Proof of Theorem.

Fix $\overline{p}$. For $a \in M$, define

$$\chi_{\overline{a}}(\overline{x}) := \bigwedge_{\substack{\psi \in \Pi_1(\overline{p}) \\ \mathcal{M} \models \psi(\overline{a})}} \psi(\overline{x}).$$

□

Outline

A Difference...

Theorem (Goncharov 1977)

There is a computable structure that is computably categorical but not relatively computably categorical.

A Difference...

Theorem (Goncharov 1977)

There is a computable structure that is computably categorical but not relatively computably categorical.

Theorem (White 2003)

The index set complexity of the computably categorical structures is Π^0_4*-hard.*

Proof of Goncharov 1977.

If the computably categorical structures and the relatively computably categorical structures coincided, then their index set complexity would coincide. This is not the case. □

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_0$ with $\mathcal{A} \models \varphi(v)$.

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_0$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_0$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.
- Wait for $\mathcal{M}_0$ to show the loop of size $3n+2$ on v.
  Restart the strategy assuming $\mathcal{A} \not\cong \mathcal{M}_0$.

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_0$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.
- Wait for $\mathcal{M}_0$ to show the loop of size $3n+2$ on v.
  Restart the strategy assuming $\mathcal{A} \not\cong \mathcal{M}_0$.
- Add loop of size $3n+1$ to u.

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_0$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.
- Wait for $\mathcal{M}_0$ to show the loop of size $3n+2$ on v.
  Restart the strategy assuming $\mathcal{A} \not\cong \mathcal{M}_0$.
- Add loop of size $3n+1$ to u.

Note that Φ_0 cannot isolate the orbits of tuples (singletons) as now $\mathcal{A} \models \varphi(u) \wedge \varphi(v)$ but u and v are not automorphic.

An Undirected Graph Example...

Theorem (Goncharov 1977)

There is a computable graph $\mathcal{G}$ that is computably categorical but not relatively computably categorical.

Proof.

In order to defeat the family Φ_0:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_0$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.
- Wait for $\mathcal{M}_0$ to show the loop of size $3n+2$ on v.
  Restart the strategy assuming $\mathcal{A} \not\cong \mathcal{M}_0$.
- Add loop of size $3n+1$ to u.

Note that Φ_0 cannot isolate the orbits of tuples (singletons) as now $\mathcal{A} \models \varphi(u) \wedge \varphi(v)$ but u and v are not automorphic.

Handling Higher Priority Strategies...

Proof (Continued...)

In order to defeat the family Φ_1:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_1$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.
- Wait for $\mathcal{M}_0$ and $\mathcal{M}_1$ to show the loop of size $3n+2$ on v. Restart the strategy assuming $\mathcal{A} \not\cong \mathcal{M}_0$ and $\mathcal{A} \not\cong \mathcal{M}_1$.
- Add loop of size $3n+1$ to u.

Note that Φ_1 cannot isolate the orbits of tuples (singletons) as now $\mathcal{A} \models \varphi(u) \wedge \varphi(v)$ but u and v are not automorphic.

Handling Higher Priority Strategies...

Proof (Continued...)

In order to defeat the family Φ_1:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_1$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.
- Wait for $\mathcal{M}_0$ and $\mathcal{M}_1$ to show the loop of size $3n+2$ on v. Restart the strategy assuming $\mathcal{A} \not\cong \mathcal{M}_0$ and $\mathcal{A} \not\cong \mathcal{M}_1$.
- Add loop of size $3n+1$ to u.

Note that Φ_1 cannot isolate the orbits of tuples (singletons) as now $\mathcal{A} \models \varphi(u) \wedge \varphi(v)$ but u and v are not automorphic.

Handling Higher Priority Strategies...

Proof (Continued...)

In order to defeat the family Φ_1:

- Build a vertex u with loop of size $3n$ and a vertex v with loops of size $3n$ and $3n+1$, where n is large.
- Wait for $\varphi \in \Phi_1$ with $\mathcal{A} \models \varphi(v)$.
- Add loop of size $3n+2$ to v.
- Wait for $\mathcal{M}_0$ and $\mathcal{M}_1$ to show the loop of size $3n+2$ on v. Restart the strategy assuming $\mathcal{A} \not\cong \mathcal{M}_0$ and $\mathcal{A} \not\cong \mathcal{M}_1$.
- Add loop of size $3n+1$ to u.

Note that Φ_1 cannot isolate the orbits of tuples (singletons) as now $\mathcal{A} \models \varphi(u) \wedge \varphi(v)$ but u and v are not automorphic.

Building the Isomorphims...

Proof (Continued...)

Assuming $\mathcal{A} \cong \mathcal{M}_i$, in order to build a computable isomorphism $\pi : \mathcal{A} \cong \mathcal{M}_i$, nonuniformly fix an isomorphism π on the part of $\mathcal{A}$ controlled by higher priority strategies. Importantly, this nonuniform part is finite.

Building the Isomorphims...

Proof (Continued...)

Assuming $\mathcal{A} \cong \mathcal{M}_i$, in order to build a computable isomorphism $\pi : \mathcal{A} \cong \mathcal{M}_i$, nonuniformly fix an isomorphism π on the part of $\mathcal{A}$ controlled by higher priority strategies. Importantly, this nonuniform part is finite.

For the remainder of the structure, be intelligent when defining the embedding π. Importantly, any embedding is necessarily an isomorphism. $\square$

Building the Isomorphims...

Proof (Continued...)

Assuming $\mathcal{A} \cong \mathcal{M}_i$, in order to build a computable isomorphism $\pi : \mathcal{A} \cong \mathcal{M}_i$, nonuniformly fix an isomorphism π on the part of $\mathcal{A}$ controlled by higher priority strategies. Importantly, this nonuniform part is finite.

For the remainder of the structure, be intelligent when defining the embedding π. Importantly, any embedding is necessarily an isomorphism. $\square$

Remark

Note that there is a (noncomputably enumerable) family Φ of Σ^0_1 formulas. Hence, the failure of relative computable categoricity is for effectiveness reasons rather than classical reasons.

Outline

An Equivalence Structure Example...

Theorem (Calvert, Cenzer, Harizanov, and Morozov 2006)

An equivalence structure is computably categorical if and only if it is relatively computably categorical if and only if there are only finitely many classes excepting classes of some fixed cardinality.

For $\alpha \geq 3$, every computable equivalence structure is relatively Δ^0_α-categorical.

An Equivalence Structure Example...

Theorem (Calvert, Cenzer, Harizanov, and Morozov 2006)

An equivalence structure is computably categorical if and only if it is relatively computably categorical if and only if there are only finitely many classes excepting classes of some fixed cardinality.

For $\alpha \geq 3$, every computable equivalence structure is relatively Δ^0_α-categorical.

Theorem (Kach and Turetsky 2009)

There is an equivalence structure that is Δ^0_2-categorical but not relatively Δ^0_2-categorical.

Useful Lemmas...

Lemma (Calvert, Cenzer, Harizanov, and Morozov 2006)

Let $\mathcal{E}$ be an equivalence structure having unbounded character (i.e., having finite classes of unbounded size) and with infinitely many infinite classes. Then $\mathcal{E}$ is not relatively Δ^0_2-categorical.

Useful Lemmas...

Lemma (Calvert, Cenzer, Harizanov, and Morozov 2006)

Let $\mathcal{E}$ be an equivalence structure having unbounded character (i.e., having finite classes of unbounded size) and with infinitely many infinite classes. Then $\mathcal{E}$ is not relatively Δ^0_2-categorical.

Lemma (Calvert, Cenzer, Harizanov, and Morozov 2006)

Let $\mathcal{E}$ be an equivalence structure for which $FIN^{\mathcal{E}}$ is intrinsically Δ^0_2-computable. Then $\mathcal{E}$ is Δ^0_2-categorical.

Building $\mathcal{E}$...

Proof.

Fix an effective enumeration $\{\mathcal{E}_i\}_{i\in\mathbb{N}}$ of the (partial) computable equivalence structures. Let $\{f_i\}_{i\in\mathbb{N}}$ be the corresponding enumeration of (total) computable functions.

Building $\mathcal{E}$...

Proof.

Fix an effective enumeration $\{\mathcal{E}_i\}_{i\in\mathbb{N}}$ of the (partial) computable equivalence structures. Let $\{f_i\}_{i\in\mathbb{N}}$ be the corresponding enumeration of (total) computable functions.

Construct $\mathcal{E}$ of unbounded character so that if $\mathcal{E} \cong \mathcal{E}_i$, then $\mathrm{FIN}^{\mathcal{E}_i}$ is Π^0_1. This is done by setting a computable threshhold for each element $x \in E_i$ and guaranteeing if $|[x]_{\mathcal{E}_i}|$ rises beyond that threshhold, then $|[x]_{\mathcal{E}_i}|$ is infinite.

Building $\mathcal{E}$...

Proof.

Fix an effective enumeration $\{\mathcal{E}_i\}_{i\in\mathbb{N}}$ of the (partial) computable equivalence structures. Let $\{f_i\}_{i\in\mathbb{N}}$ be the corresponding enumeration of (total) computable functions.

Construct $\mathcal{E}$ of unbounded character so that if $\mathcal{E} \cong \mathcal{E}_i$, then $\mathrm{FIN}^{\mathcal{E}_i}$ is Π^0_1. This is done by setting a computable threshhold for each element $x \in E_i$ and guaranteeing if $|[x]_{\mathcal{E}_i}|$ rises beyond that threshhold, then $|[x]_{\mathcal{E}_i}|$ is infinite.

At stage $s > 0$, operate in three steps:

Building $\mathcal{E}$...

Proof.

Fix an effective enumeration $\{\mathcal{E}_i\}_{i\in\mathbb{N}}$ of the (partial) computable equivalence structures. Let $\{f_i\}_{i\in\mathbb{N}}$ be the corresponding enumeration of (total) computable functions.

Construct $\mathcal{E}$ of unbounded character so that if $\mathcal{E} \cong \mathcal{E}_i$, then $\mathrm{FIN}^{\mathcal{E}_i}$ is Π^0_1. This is done by setting a computable threshhold for each element $x \in E_i$ and guaranteeing if $|[x]_{\mathcal{E}_i}|$ rises beyond that threshhold, then $|[x]_{\mathcal{E}_i}|$ is infinite.

At stage $s > 0$, operate in three steps:

1. Force any class in $\mathcal{E}$ of size $f_i(n, s)$ with $i, n < s$ for which $f_i(n, s) > 2^{i+n}$ to become infinite.

Building $\mathcal{E}$...

Proof.

Fix an effective enumeration $\{\mathcal{E}_i\}_{i\in\mathbb{N}}$ of the (partial) computable equivalence structures. Let $\{f_i\}_{i\in\mathbb{N}}$ be the corresponding enumeration of (total) computable functions.

Construct $\mathcal{E}$ of unbounded character so that if $\mathcal{E} \cong \mathcal{E}_i$, then $\mathrm{FIN}^{\mathcal{E}_i}$ is Π^0_1. This is done by setting a computable threshhold for each element $x \in E_i$ and guaranteeing if $|[x]_{\mathcal{E}_i}|$ rises beyond that threshhold, then $|[x]_{\mathcal{E}_i}|$ is infinite.

At stage $s > 0$, operate in three steps:

1. Force any class in $\mathcal{E}$ of size $f_i(n, s)$ with $i, n < s$ for which $f_i(n, s) > 2^{i+n}$ to become infinite.
2. Build in $\mathcal{E}$, if one does not already exist, a class of size k for each $k < s$ not within the set $\{f_i(n, s) : i, n < s \text{ and } f_i(n, s) > 2^{i+n}\}$.

Building $\mathcal{E}$...

Proof.

Fix an effective enumeration $\{\mathcal{E}_i\}_{i\in\mathbb{N}}$ of the (partial) computable equivalence structures. Let $\{f_i\}_{i\in\mathbb{N}}$ be the corresponding enumeration of (total) computable functions.

Construct $\mathcal{E}$ of unbounded character so that if $\mathcal{E} \cong \mathcal{E}_i$, then $\mathrm{FIN}^{\mathcal{E}_i}$ is Π^0_1. This is done by setting a computable threshhold for each element $x \in E_i$ and guaranteeing if $|[x]_{\mathcal{E}_i}|$ rises beyond that threshhold, then $|[x]_{\mathcal{E}_i}|$ is infinite.

At stage $s > 0$, operate in three steps:

1. Force any class in $\mathcal{E}$ of size $f_i(n, s)$ with $i, n < s$ for which $f_i(n, s) > 2^{i+n}$ to become infinite.
2. Build in $\mathcal{E}$, if one does not already exist, a class of size k for each $k < s$ not within the set $\{f_i(n, s) : i, n < s \text{ and } f_i(n, s) > 2^{i+n}\}$.
3. Create a new infinite equivalence class in $\mathcal{E}$.

Building $\mathcal{E}$...

Proof.

- Force any class in $\mathcal{E}$ of size $f_i(n,s)$ with $i,n<s$ for which $f_i(n,s)>2^{i+n}$ to become infinite.
- Build in $\mathcal{E}$, if one does not already exist, a class of size k for each $k<s$ not within the set $\{f_i(n,s) : i,n<s \text{ and } f_i(n,s)>2^{i+n}\}$.

Claim

The structure $\mathcal{E}$ has unbounded character.

Proof.

Fixing a positive integer k, there are at most $(1+\log k)^2$ many pairs (i,n) with $2^{i+n}\leq k$. Thus, at any stage s,

$$\left|\left\{m\leq k : (\exists i,n<s)\left[m=f_i(n,s)>2^{i+n}\right]\right\}\right|\leq(1+\log k)^2$$

Note that $\lim_{k\to\infty}(k-(1+\log k)^2)=\infty$. □

Building $\mathcal{E}$...

Proof.

- Force any class in $\mathcal{E}$ of size $f_i(n, s)$ with $i, n < s$ for which $f_i(n, s) > 2^{i+n}$ to become infinite.
- Build in $\mathcal{E}$, if one does not already exist, a class of size k for each $k < s$ not within the set $\{f_i(n, s) : i, n < s \text{ and } f_i(n, s) > 2^{i+n}\}$.

Claim

If $\mathcal{E} \cong \mathcal{E}_i$, then $\mathrm{FIN}^{\mathcal{E}_i}$ is Π^0_1.

Proof.

Note that $x \in \mathrm{FIN}^{\mathcal{E}_i}$ if $(\forall s)\left[f_i(x, s) < 2^{i+x}\right]$.

Conversely, if $x \in \mathrm{FIN}^{\mathcal{E}_i}$, then either $F_i(x) \leq 2^{i+x}$ or $F_i(x) > 2^{i+x}$.

1. If the former, then $(\forall s)\left[f_i(x, s) \leq 2^{i+x}\right]$.
2. If the latter, then $\mathcal{E} \not\cong \mathcal{E}_i$ as $\mathcal{E}$ has no class of size $F_i(x)$. $\square$

The Reason for the Failure...

Remark

Note that, in contrast to the previous example, there is no classical family Φ of Σ_2^c formulas. Hence, the failure of relative computable categoricity is for classical reasons rather than effectiveness reasons.

Outline

Arithmetic and Hyperarithmetic...

Definition

A computable structure $\mathcal{M}$ is *arithmetically categorical* if between any two computable presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is an arithmetic isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$.

Definition

A computable structure $\mathcal{M}$ is *relatively arithmetically categorical* if between any presentations $\mathcal{A}$ and $\mathcal{B}$ of $\mathcal{M}$ there is an $(X \oplus Y)$-computable isomorphism $\pi : \mathcal{A} \cong \mathcal{B}$, where X is arithmetic in $\mathcal{A}$ and Y is arithmetic in $\mathcal{B}$.

The Importance of Being Relative...

Theorem (Ash 1987)

Any relatively arithmetically categorical structure is relatively Δ^0_n-categorical for some integer $n \in \mathbb{N}$.

Theorem (Kach and Montalbán)

There is a computable structure $\mathcal{M}$ that is arithmetically categorical but not Δ^0_n-categorical for any integer $n \in \mathbb{N}$.

The Language...

Proof.

The structure is in the language $\mathcal{L}$ consisting of:

The Language...

Proof.

The structure is in the language $\mathcal{L}$ consisting of:

- $\{C_k\}_{k \in \mathbb{N}}$: Unary *component relations*, specifying whether an element $x \in M$ is in the kth component (tree).

The Language...

Proof.

The structure is in the language $\mathcal{L}$ consisting of:

- $\{C_k\}_{k\in\mathbb{N}}$: Unary *component relations*, specifying whether an element $x \in M$ is in the kth component (tree).
- $\{E\}$: Binary *edge relation*, specifying the edge relation on each tree.

The Language...

Proof.

The structure is in the language $\mathcal{L}$ consisting of:

- $\{\mathrm{C}_k\}_{k\in\mathbb{N}}$: Unary *component relations*, specifying whether an element $x \in M$ is in the kth component (tree).
- $\{\mathrm{E}\}$: Binary *edge relation*, specifying the edge relation on each tree.
- $\{\mathrm{M}_j\}_{j\in\mathbb{N}}$: Unary *marker relations*, specifying whether a leaf node is marked by j.

The Language...

Proof.

The structure is in the language $\mathcal{L}$ consisting of:

- $\{C_k\}_{k\in\mathbb{N}}$: Unary *component relations*, specifying whether an element $x \in M$ is in the kth component (tree).
- $\{E\}$: Binary *edge relation*, specifying the edge relation on each tree.
- $\{M_j\}_{j\in\mathbb{N}}$: Unary *marker relations*, specifying whether a leaf node is marked by j.
- $\{L_i\}_{i\in\mathbb{N}}$: Unary *temporary labels*, identifying an element $x \in M$ uniquely at a **specific** stage in the construction.

The Language...

Proof.

The structure is in the language $\mathcal{L}$ consisting of:

- $\{\mathrm{C}_k\}_{k\in\mathbb{N}}$: Unary *component relations*, specifying whether an element $x \in M$ is in the kth component (tree).
- $\{\mathrm{E}\}$: Binary *edge relation*, specifying the edge relation on each tree.
- $\{\mathrm{M}_j\}_{j\in\mathbb{N}}$: Unary *marker relations*, specifying whether a leaf node is marked by j.
- $\{\mathrm{L}_i\}_{i\in\mathbb{N}}$: Unary *temporary labels*, identifying an element $x \in M$ uniquely at a **specific** stage in the construction.

The last relation is not only subtle, but also *computably enumerable*. More on it later... The other relations are straightforward.

The Basic Trees...

Definition

A *term* is a pair $(k, m) \in \mathbb{N}^2$. The *rank* of a term (k, m) is k. The *marker* of a term $(0, m)$ is m.

The Basic Trees...

Definition

A *term* is a pair $(k, m) \in \mathbb{N}^2$. The *rank* of a term (k, m) is k. The *marker* of a term $(0, m)$ is m.

A nonempty sequence of terms $\langle (k_1, m_1), \ldots, (k_\ell, m_\ell) \rangle$ is *acceptable* if $k_i = k_{i+1} + 1$ and $m_i \neq m_{i+1}$ for $1 \leq i < \ell$.

The Basic Trees...

Definition

A *term* is a pair $(k, m) \in \mathbb{N}^2$. The *rank* of a term (k, m) is k. The *marker* of a term $(0, m)$ is m.

A nonempty sequence of terms $\langle (k_1, m_1), \ldots, (k_\ell, m_\ell) \rangle$ is *acceptable* if $k_i = k_{i+1} + 1$ and $m_i \neq m_{i+1}$ for $1 \leq i < \ell$.

Definition

Fix a term (k, m). The tree $\mathrm{T}_{(k,m)}$ is the set of acceptable strings σ with $\sigma(0) = t$.

The Basic Trees...

Definition

A *term* is a pair $(k, m) \in \mathbb{N}^2$. The *rank* of a term (k, m) is k. The *marker* of a term $(0, m)$ is m.

A nonempty sequence of terms $\langle (k_1, m_1), \ldots, (k_\ell, m_\ell) \rangle$ is *acceptable* if $k_i = k_{i+1} + 1$ and $m_i \neq m_{i+1}$ for $1 \leq i < \ell$.

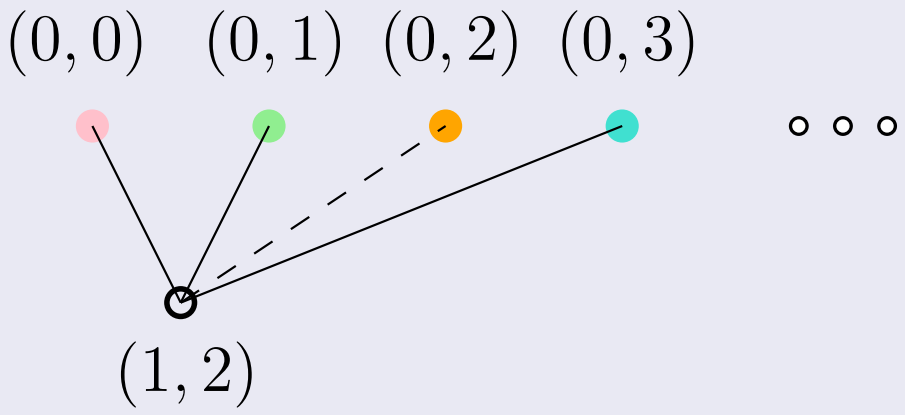
Definition

Fix a term (k, m). The tree $\mathrm{T}_{(k,m)}$ is the set of acceptable strings σ with $\sigma(0) = t$.

The structure $\mathcal{T}_{(k,m)}$ is the tree $\mathrm{T}_{(k,m)}$ with the component relation C_k, its edge relation E, and the marker relations M_m.

Pictures...

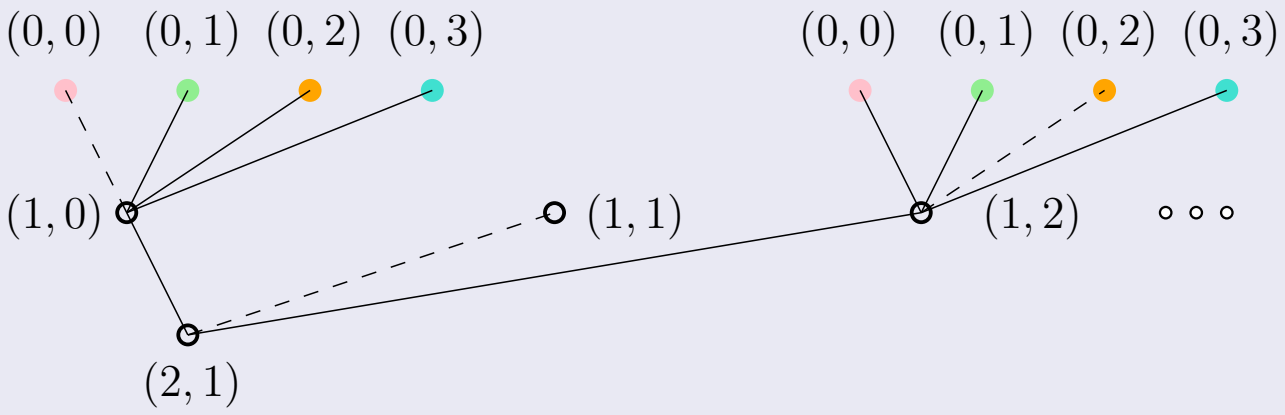
Example



The Basic Tree $\mathcal{T}_{(1,2)}$

Pictures...

Example



The Basic Tree $\mathcal{T}_{(2,1)}$

The Intuition for Not Δ^0_n-Categoricity...

Lemma

Fix integers $k > 0$ and $m \in \mathbb{N}$. The tree $\mathcal{T}_{(k,m)}$ is not Δ^0_{k-1}-categorical.

The Intuition for Not Δ^0_n-Categoricity...

Lemma

Fix integers $k > 0$ and $m \in \mathbb{N}$. The tree $\mathcal{T}_{(k,m)}$ is not Δ^0_{k-1}-categorical.

Lemma (Ash and Knight 2000)

Fix an integer $n > 0$. Let $\mathcal{A}$ be a computable presentation of a computable structure. If:

- *The presentation $\mathcal{A}$ is $(n-1)$-friendly.*
- *The back-and-forth relation $\not\leq_n$ is computably enumerable.*
- *For every tuple $\overline{c} \in A$, there is a tuple $\overline{a} \in A$ that is n-free over $\overline{c}$.*

Then $\mathcal{A}$ is not Δ^0_n-categorical.

Back-and-Forth Relations...

Definition

Fix a computable ordinal α. Define the αth back-and-forth relation $\leq_\alpha$ by $\mathcal{A} \leq_\alpha \mathcal{B}$ if

$$\Pi^c_\alpha - \mathrm{Th}(\mathcal{A}) \subseteq \Pi^c_\alpha - \mathrm{Th}(\mathcal{B}).$$

For $\overline{a}, \overline{a}' \in A$, define $\overline{a} \leq_\alpha \overline{a}'$ if $(\mathcal{A}; \overline{a}) \leq_\alpha (\mathcal{A}; \overline{a}')$.

Back-and-Forth Relations...

Definition

Fix a computable ordinal α. Define the αth back-and-forth relation $\leq_\alpha$ by $\mathcal{A} \leq_\alpha \mathcal{B}$ if

$$\Pi^c_\alpha - \mathrm{Th}(\mathcal{A}) \subseteq \Pi^c_\alpha - \mathrm{Th}(\mathcal{B}).$$

For $\overline{a}, \overline{a}' \in A$, define $\overline{a} \leq_\alpha \overline{a}'$ if $(\mathcal{A}; \overline{a}) \leq_\alpha (\mathcal{A}; \overline{a}')$.

Definition

A computable presentation $\mathcal{A}$ is α-friendly if the back-and-forth relations $\leq_\beta$, for $\beta < \alpha$, are uniformly computably enumerable.

Back-and-Forth Relations...

Definition

Fix a computable ordinal α. Define the αth back-and-forth relation $\leq_\alpha$ by $\mathcal{A} \leq_\alpha \mathcal{B}$ if

$$\Pi^c_\alpha - \mathrm{Th}(\mathcal{A}) \subseteq \Pi^c_\alpha - \mathrm{Th}(\mathcal{B}).$$

For $\overline{a}, \overline{a}' \in A$, define $\overline{a} \leq_\alpha \overline{a}'$ if $(\mathcal{A}; \overline{a}) \leq_\alpha (\mathcal{A}; \overline{a}')$.

Definition

A computable presentation $\mathcal{A}$ is α-friendly if the back-and-forth relations $\leq_\beta$, for $\beta < \alpha$, are uniformly computably enumerable.

Definition

A tuple $\overline{a} \in A$ is α-free over $\overline{c} \in A$ if for all $\overline{b} \in A$ and $\beta < \alpha$, there exist $\overline{a}' \in A$ and $\overline{b}' \in A$ such that $\overline{c}\,\overline{a}\,\overline{b} \leq_\beta \overline{c}\,\overline{a}'\,\overline{b}'$ and $\overline{c}\,\overline{a}' \not\leq_\alpha \overline{c}\,\overline{a}$.

Not Δ^0_n-Categorical...

Lemma

Fix integers $k > 0$ and $m \in \mathbb{N}$. The tree $\mathcal{T}_{(k,m)}$ is not Δ^0_{k-1}-categorical.

Proof.

The presentation $\mathcal{A}$ is $(n-1)$-friendly: The back-and-forth relations are computable for the standard presentation.

The back-and-forth relation $\not\leq_n$ is computably enumerable: The back-and-forth relations are computable for the standard presentation.

For every tuple $\overline{c} \in A$, there is a tuple $\overline{a} \in A$ that is n-free over $\overline{c}$: Choose (singleton) a away from $\overline{c}$. Fixing $\overline{b}$, choose a' away from a, $\overline{b}$, and $\overline{c}$. Let $\overline{b}'$ be in the same position with respect to a' as $\overline{b}$ is with respect to a. $\square$

Temporary Labels...

Remark

Though seemingly counter-intuitive, we discuss how to make $\mathcal{T}_{(2,1)}$, for example, computably categorical. The construction maintains a global *bag of (temporary) labels*, a subset of $\{\mathrm{L}_i\}_{i\in\mathbb{N}}$.

Temporary Labels...

Remark

Though seemingly counter-intuitive, we discuss how to make $\mathcal{T}_{(2,1)}$, for example, computably categorical. The construction maintains a global *bag of (temporary) labels*, a subset of $\{\mathrm{L}_i\}_{i\in\mathbb{N}}$.

At each stage of the construction, every element $x \in \mathrm{T}_{(2,1)}$ has unique temporary labels. These computably identify x in $\mathcal{M}_i$ if $\mathcal{A} \cong \mathcal{M}_i$.

Temporary Labels...

Remark

Though seemingly counter-intuitive, we discuss how to make $\mathcal{T}_{(2,1)}$, for example, computably categorical. The construction maintains a global *bag of (temporary) labels*, a subset of $\{\mathrm{L}_i\}_{i\in\mathbb{N}}$.

At each stage of the construction, every element $x \in \mathrm{T}_{(2,1)}$ has unique temporary labels. These computably identify x in $\mathcal{M}_i$ if $\mathcal{A} \cong \mathcal{M}_i$. After isomorphisms catch up on x, we:

Temporary Labels...

Remark

Though seemingly counter-intuitive, we discuss how to make $\mathcal{T}_{(2,1)}$, for example, computably categorical. The construction maintains a global *bag of (temporary) labels*, a subset of $\{\mathrm{L}_i\}_{i\in\mathbb{N}}$.

At each stage of the construction, every element $x \in \mathrm{T}_{(2,1)}$ has unique temporary labels. These computably identify x in $\mathcal{M}_i$ if $\mathcal{A} \cong \mathcal{M}_i$. After isomorphisms catch up on x, we:

- Enumerate the unique temporary labels on x into the bag of labels (making them no longer unique).
- Apply all the temporary labels in the bag of labels to x.
- Apply new unique temporary labels to x.

Temporary Labels...

Remark

Though seemingly counter-intuitive, we discuss how to make $\mathcal{T}_{(2,1)}$, for example, computably categorical. The construction maintains a global *bag of (temporary) labels*, a subset of $\{\mathrm{L}_i\}_{i\in\mathbb{N}}$.

At each stage of the construction, every element $x \in \mathrm{T}_{(2,1)}$ has unique temporary labels. These computably identify x in $\mathcal{M}_i$ if $\mathcal{A} \cong \mathcal{M}_i$. After isomorphisms catch up on x, we:

- Enumerate the unique temporary labels on x into the bag of labels (making them no longer unique).
- Apply all the temporary labels in the bag of labels to x.
- Apply new unique temporary labels to x.

Recall that the temporary labels were *computably enumerable* rather than computable.

A (Major) Issue...

Remark

The issue, of course, is that isomorphisms will not necessarily catch up. This causes Σ^0_1-differences that would break the n-freeness requirement.

A (Major) Issue...

Remark

The issue, of course, is that isomorphisms will not necessarily catch up. This causes Σ^0_1-differences that would break the *n*-freeness requirement.

Consequently, the tree of strategies must guess which isomorphisms will catch up and which will not. Any *garbage* constructed off the true path must be distributed across the tree.

A (Major) Issue...

Remark

The issue, of course, is that isomorphisms will not necessarily catch up. This causes Σ^0_1-differences that would break the n-freeness requirement.

Consequently, the tree of strategies must guess which isomorphisms will catch up and which will not. Any *garbage* constructed off the true path must be distributed across the tree.

Unfortunately, unlike in the earlier example, an individual strategy must build infinitely much stuff. This causes the earlier computable categoricity argument to fail as it exploited the necessity of only finitely much nonuniform information.

A Fix...

Remark

As a workaround, we attempt to define an isomorphism modulo certain permutations of nodes. This necessitates a complicated set of outcomes for the isomorphism requirement for $\mathcal{M}_i$:

$$\mathtt{pe}_i < \mathtt{pe}_{i-1} < \cdots < \mathtt{pe}_1 < \mathtt{pe}_0 < \mathtt{se} < \mathtt{f}$$

A Fix...

Remark

As a workaround, we attempt to define an isomorphism modulo certain permutations of nodes. This necessitates a complicated set of outcomes for the isomorphism requirement for $\mathcal{M}_i$:

$$\mathtt{pe}_i < \mathtt{pe}_{i-1} < \cdots < \mathtt{pe}_1 < \mathtt{pe}_0 < \mathtt{se} < \mathtt{f}$$

In fact, life is even more complicated than this because of the need to never revisit a *dead* node (reinitialization is worse and avoided).

A Fix...

Remark

As a workaround, we attempt to define an isomorphism modulo certain permutations of nodes. This necessitates a complicated set of outcomes for the isomorphism requirement for $\mathcal{M}_i$:

$$\mathtt{pe}_i < \mathtt{pe}_{i-1} < \cdots < \mathtt{pe}_1 < \mathtt{pe}_0 < \mathtt{se} < \mathtt{f}$$

In fact, life is even more complicated than this because of the need to never revisit a *dead* node (reinitialization is worse and avoided).

These outcomes measure how isomorphic $\mathcal{A}$ and $\mathcal{M}_i$ appear to be. The $\mathtt{f}$ outcomes indicates not at all, the $\mathtt{pe}_i$ outcome indicates as much as possible, and the intermediate outcomes indicate varying extents.

A Fix...

Remark

As a workaround, we attempt to define an isomorphism modulo certain permutations of nodes. This necessitates a complicated set of outcomes for the isomorphism requirement for $\mathcal{M}_i$:

$$\mathtt{pe}_i < \mathtt{pe}_{i-1} < \cdots < \mathtt{pe}_1 < \mathtt{pe}_0 < \mathtt{se} < \mathtt{f}$$

In fact, life is even more complicated than this because of the need to never revisit a *dead* node (reinitialization is worse and avoided).

These outcomes measure how isomorphic $\mathcal{A}$ and $\mathcal{M}_i$ appear to be. The $\mathtt{f}$ outcomes indicates not at all, the $\mathtt{pe}_i$ outcome indicates as much as possible, and the intermediate outcomes indicate varying extents.

The important point is that the true outcome will be $\mathtt{pe}_i$ if $\mathcal{A} \cong \mathcal{M}_i$ (though not necessarily conversely).

Arithmetic Categoricity and Not Δ^0_n-Categoricity...

Remark

Waving away the details of computable categoricity, there is still the issue that we want only arithmetic categoricity. Thus, when waiting for isomorphisms between $\mathcal{A}$ and $\mathcal{M}_i$ to catch up, we ignore $\bigcup_{k<i} \mathrm{C}_k$.

Arithmetic Categoricity and Not Δ^0_n-Categoricity...

Remark

Waving away the details of computable categoricity, there is still the issue that we want only arithmetic categoricity. Thus, when waiting for isomorphisms between $\mathcal{A}$ and $\mathcal{M}_i$ to catch up, we ignore $\bigcup_{k<i} \mathsf{C}_k$.

Arithmetic Categoricity: If we can computably build an isomorphism

$$\pi : \mathcal{A} \upharpoonright \bigcup_{k \geq i} \mathsf{C}_k \cong \mathcal{M}_i \upharpoonright \bigcup_{k \geq i} \mathsf{C}_k,$$

then there is a Δ^0_{i+1}-isomorphism $\pi : \mathcal{A} \cong \mathcal{M}_i$.

Arithmetic Categoricity and Not Δ^0_n-Categoricity...

Remark

Waving away the details of computable categoricity, there is still the issue that we want only arithmetic categoricity. Thus, when waiting for isomorphisms between $\mathcal{A}$ and $\mathcal{M}_i$ to catch up, we ignore $\bigcup_{k<i} \mathsf{C}_k$.

Arithmetic Categoricity: If we can computably build an isomorphism

$$\pi : \mathcal{A} \upharpoonright \bigcup_{k \geq i} \mathsf{C}_k \cong \mathcal{M}_i \upharpoonright \bigcup_{k \geq i} \mathsf{C}_k,$$

then there is a Δ^0_{i+1}-isomorphism $\pi : \mathcal{A} \cong \mathcal{M}_i$.

Not Δ^0_n-Categoricity: Since only finitely many requirements influence any C_k, their action is essentially (nonuniformly) computable. Thus, this property is preserved from the basic tree.

Outline

Feferman and Spector's $\mathcal{O}^*$...

Definition (Feferman and Spector 1962)

There is a partial order $\mathcal{O}^* = (\mathcal{O}^*; \preccurlyeq)$ with the $\preccurlyeq$-relation c.e. and:

- For all $\alpha \in \mathcal{O}^*$, the set $\{\beta \preccurlyeq \alpha\}$ is linearly ordered and has no infinite hyperarithmetic descending sequence.
- The set $\mathcal{O}^*$ has a $\preccurlyeq$-least element. The set of successor and limit elements and the predecessor function are computable.
- The set of $\alpha \in \mathcal{O}^*$ for which $\{\beta \in \mathcal{O}^* : \beta \preccurlyeq \alpha\}$ is well-ordered is isomorphic to $\mathcal{O}$.
- There is a computable sequence $\{\alpha_n \in \mathcal{O}^* : n \in \mathbb{N}\}$ such that the set $\{n \in \mathbb{N} : \alpha_n \in \mathcal{O}\}$ is Π^1_1-complete.

Feferman and Spector's $\mathcal{O}^*$...

Definition (Feferman and Spector 1962)

There is a partial order $\mathcal{O}^* = (\mathcal{O}^*; \preccurlyeq)$ with the $\preccurlyeq$-relation c.e. and:

- For all $\alpha \in \mathcal{O}^*$, the set $\{\beta \preccurlyeq \alpha\}$ is linearly ordered and has no infinite hyperarithmetic descending sequence.
- The set $\mathcal{O}^*$ has a $\preccurlyeq$-least element. The set of successor and limit elements and the predecessor function are computable.
- The set of $\alpha \in \mathcal{O}^*$ for which $\{\beta \in \mathcal{O}^* : \beta \preccurlyeq \alpha\}$ is well-ordered is isomorphic to $\mathcal{O}$.
- There is a computable sequence $\{\alpha_n \in \mathcal{O}^* : n \in \mathbb{N}\}$ such that the set $\{n \in \mathbb{N} : \alpha_n \in \mathcal{O}\}$ is Π^1_1-complete.

Remark

It does little harm to imagine $\mathcal{O}^*$ as being a computable presentation of $\omega_1^{CK} \cdot (1 + \eta)$ with no hyperarithmetic descending sequences and with computable successor and limit elements and predecessor function.

The Big Theorem...

Theorem (Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky)

The index set complexity of the computably categorical structures is Π^1_1-complete.

Proof.

Fix $\{\alpha_n\}_{n\in\mathbb{N}}$ such that $\{n \in \mathbb{N} : \alpha_n \in \mathcal{O}\}$ is Π^1_1-complete. Build, uniformly in $\alpha \in \mathcal{O}^*$, a computable structure $\mathcal{A}_\alpha$, where $\mathcal{A}_\alpha$ is computably categorical if and only if $\alpha \in \mathcal{O}$. $\square$

The Big Theorem...

Theorem (Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky)

The index set complexity of the computably categorical structures is Π^1_1-complete.

Proof.

Fix $\{\alpha_n\}_{n\in\mathbb{N}}$ such that $\{n \in \mathbb{N} : \alpha_n \in \mathcal{O}\}$ is Π^1_1-complete. Build, uniformly in $\alpha \in \mathcal{O}^*$, a computable structure $\mathcal{A}_\alpha$, where $\mathcal{A}_\alpha$ is computably categorical if and only if $\alpha \in \mathcal{O}$. $\square$

Corollary

For each computable ordinal α, there is a computably categorical structure that is not relatively Δ^0_α categorical.

Space in the Margins...

Theorem (Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky)

Fix a computable ordinal α. There is a computable structure $\mathcal{S}$ that is computably categorical but not relatively Δ^0_α-categorical.

Space in the Margins...

Theorem (Downey, Kach, Lempp, Lewis, Montalbán, and Turetsky)

Fix a computable ordinal α. There is a computable structure $\mathcal{S}$ that is computably categorical but not relatively Δ^0_α-categorical.

Proof.

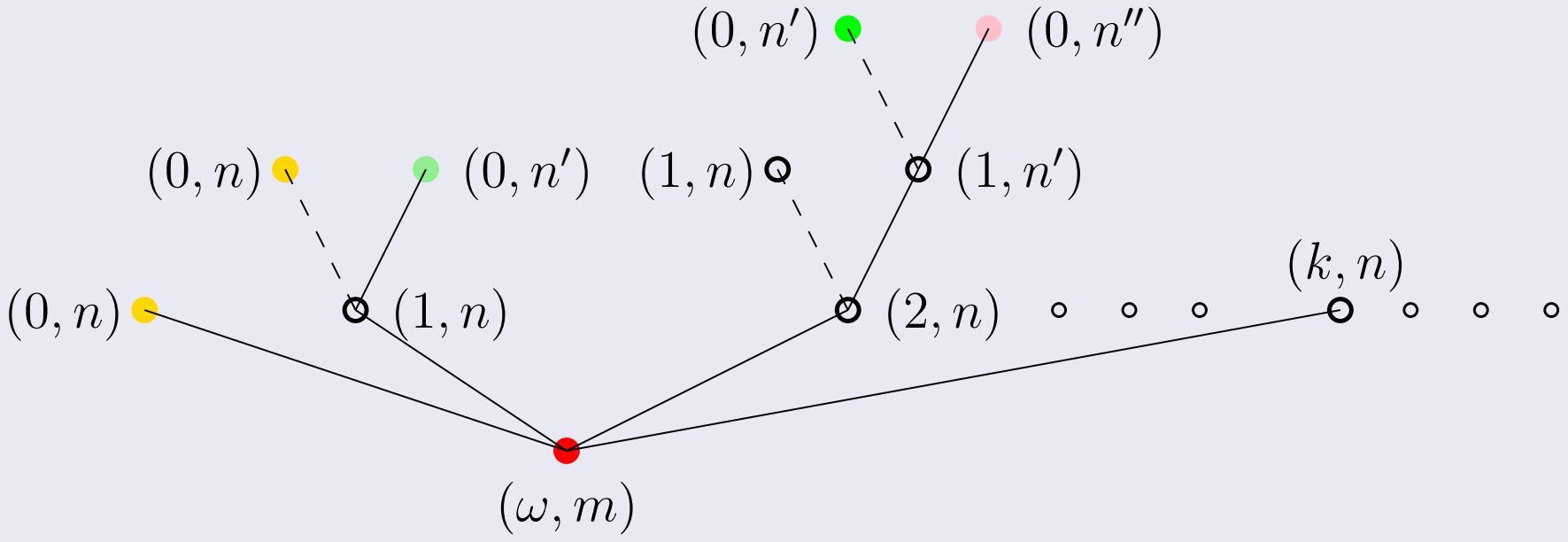
Maintain computable categoricity by, at every stage, identifying vertices with a (temporary) property not shared anywhere else.

Prevent relative Δ^0_α-categoricity by preventing any Σ^i_α Scott Family. Do so by using (infinitely-branching) trees of height roughly α. $\square$

The Basic Trees...

Remark

For $\alpha = \omega$, we illustrate the basic trees $\mathrm{T}_{(\omega,m)}$ and $\mathrm{T}_{(\omega,m,(k,n))}$.

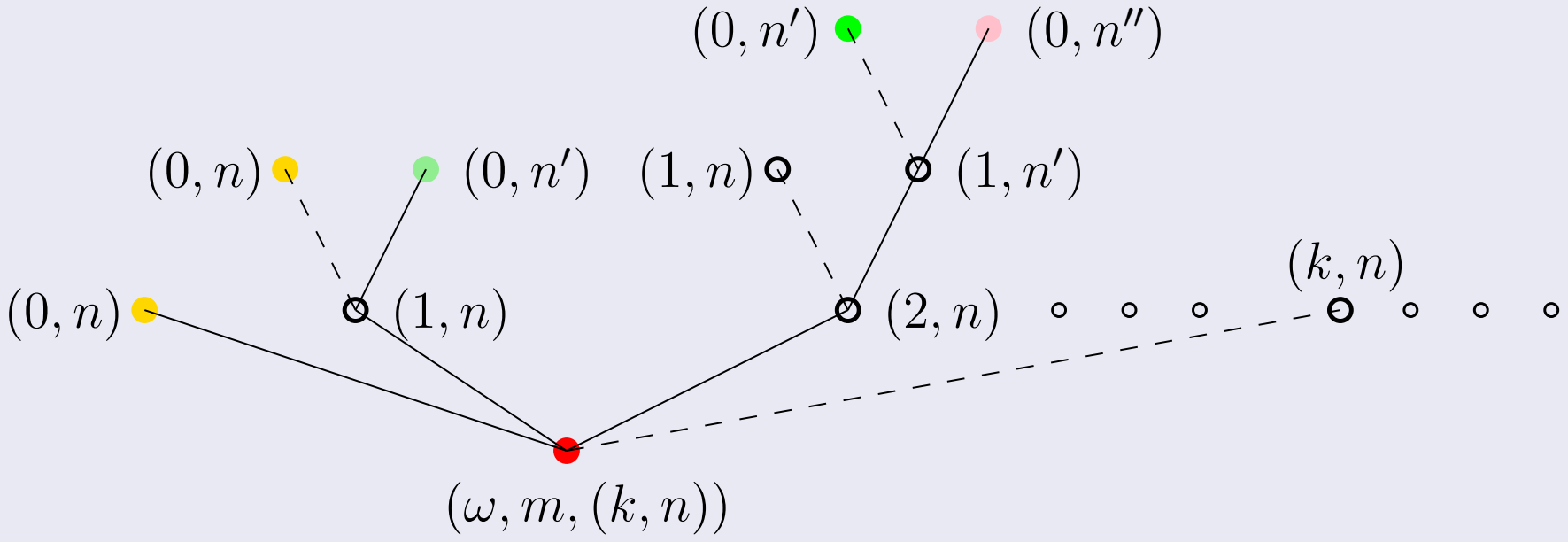


The Basic Tree $\mathrm{T}_{(\omega,m)}$

The Basic Trees...

Remark

For $\alpha = \omega$, we illustrate the basic trees $\mathrm{T}_{(\omega,m)}$ and $\mathrm{T}_{(\omega,m,(k,n))}$.



The Basic Tree $\mathrm{T}_{(\omega,m,(k,n))}$

References

C. J. Ash.
Categoricity in hyperarithmetical degrees.
Ann. Pure Appl. Logic, 34(1):1–14, 1987.

C. J. Ash and J. Knight.
Computable structures and the hyperarithmetical hierarchy, volume 144 of *Studies in Logic and the Foundations of Mathematics*.
North-Holland Publishing Co., Amsterdam, 2000.

Wesley Calvert, Douglas Cenzer, Valentina Harizanov, and Andrei Morozov.
Effective categoricity of equivalence structures.
Ann. Pure Appl. Logic, 141(1-2):61–78, 2006.

R. Douni, D. Khirshvel′d, and B. Khusainov.
Uniformity in the theory of computable structures.
Algebra Logika, 42(5):566–593, 637, 2003.

Rodney G. Downey, Asher M. Kach, Steffen Lempp, and Daniel D. Turetsky.
Computable categoricity versus relative computable categoricity.
submitted.

S. S. Gončarov.
Selfstability, and computable families of constructivizations.
Algebra i Logika, 14(6):647–680, 727, 1975.

S. S. Gončarov.
The number of nonautoequivalent constructivizations.
Algebra i Logika, 16(3):257–282, 377, 1977.

Asher M. Kach and Daniel Turetsky.
Δ^0_2-categoricity of equivalence structures.
New Zealand J. Math., 39:143–149, 2009.