Computable categoricity relative to a degree

SIU Online Logic Seminar

Java Darleen Villano

October 24th, 2024

University of Connecticut

Outline

1. Notions of categoricity
2. Categoricity relative to a degree
   - Above $\mathbf{0}''$ and below $\mathbf{0}'$
   - Generalizing the DHM result
3. Extensions of current work
   - Embedding lattices
   - Categoricity relative to a generic degree
   - Focusing on structures
4. Proof sketch of the poset result

Notions of categoricity

Definitions

Definition

A computable structure $\mathcal{A}$ is **computably categorical** if for every computable copy $\mathcal{B}$ of $\mathcal{A}$, there exists a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Definitions

Definition

A computable structure $\mathcal{A}$ is **computably categorical** if for every computable copy $\mathcal{B}$ of $\mathcal{A}$, there exists a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Definition

A computable structure $\mathcal{A}$ is **relatively computably categorical** if for every copy (not necessarily computable) $\mathcal{B}$ of $\mathcal{A}$, there is a $\mathcal{B}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Definitions

Definition

A computable structure $\mathcal{A}$ is **computably categorical** if for every computable copy $\mathcal{B}$ of $\mathcal{A}$, there exists a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Definition

A computable structure $\mathcal{A}$ is **relatively computably categorical** if for every copy (not necessarily computable) $\mathcal{B}$ of $\mathcal{A}$, there is a $\mathcal{B}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

These notions are not equivalent in general.

Definitions

Definition

A computable structure $\mathcal{A}$ is **computably categorical** if for every computable copy $\mathcal{B}$ of $\mathcal{A}$, there exists a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Definition

A computable structure $\mathcal{A}$ is **relatively computably categorical** if for every copy (not necessarily computable) $\mathcal{B}$ of $\mathcal{A}$, there is a $\mathcal{B}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

These notions are not equivalent in general. Gončarov [Gon77] built the first example of a structure which was computably categorical but *not* relatively computably categorical.

Relativizing categoricity

The following relativization of categoricity appears in [DHTM21].

Relativizing categoricity

The following relativization of categoricity appears in [DHTM21].

Definition

For a Turing degree $\mathbf{d}$, a computable structure $\mathcal{A}$ is **computably categorical relative to d** if for every $\mathbf{d}$-computable copy $\mathcal{B}$ of $\mathcal{A}$, there is a $\mathbf{d}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Relativizing categoricity

The following relativization of categoricity appears in [DHTM21].

Definition

For a Turing degree $\mathbf{d}$, a computable structure $\mathcal{A}$ is **computably categorical relative to d** if for every $\mathbf{d}$-computable copy $\mathcal{B}$ of $\mathcal{A}$, there is a $\mathbf{d}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

This is distinct from being $\mathbf{d}$-computably categorical.

Definition

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

Relativizing categoricity

The following relativization of categoricity appears in [DHTM21].

Definition

For a Turing degree **d**, a computable structure $\mathcal{A}$ is **computably categorical relative to d** if for every **d**-computable copy $\mathcal{B}$ of $\mathcal{A}$, there is a **d**-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

This is distinct from being **d**-computably categorical.

Definition

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

Relativizing categoricity

Definition

For a Turing degree **d**, a computable structure $\mathcal{A}$ is **computably categorical relative to d** if for every **d**-computable copy $\mathcal{B}$ of $\mathcal{A}$, there is a **d**-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

It is also distinct from being relatively Δ^0_α-categorical.

Relativizing categoricity

Definition

For a Turing degree $\mathbf{d}$, a computable structure $\mathcal{A}$ is **computably categorical relative to d** if for every $\mathbf{d}$-computable copy $\mathcal{B}$ of $\mathcal{A}$, there is a $\mathbf{d}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

It is also distinct from being relatively Δ^0_α-categorical.

Definition

A computable structure $\mathcal{A}$ is **relatively Δ^0_α-categorical** if for any copy $\mathcal{B}$ of $\mathcal{A}$, there is a $\Delta^0_\alpha(\mathcal{B})$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Relativizing categoricity

Definition

For a Turing degree $\mathbf{d}$, a computable structure $\mathcal{A}$ is **computably categorical relative to d** if for every $\mathbf{d}$-computable copy $\mathcal{B}$ of $\mathcal{A}$, there is a $\mathbf{d}$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

It is also distinct from being relatively Δ^0_α-categorical.

Definition

A computable structure $\mathcal{A}$ is **relatively Δ^0_α-categorical** if for any copy $\mathcal{B}$ of $\mathcal{A}$, there is a $\Delta^0_\alpha(\mathcal{B})$-computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

Categoricity relative to a degree

Background

How does computable categoricity relative to a degree behave?

Background

How does computable categoricity relative to a degree behave?

Fact

A computable structure $\mathcal{A}$ is **relatively computably categorical** *if for all degrees* $\mathbf{d}$*,* $\mathcal{A}$ *is computably categorical relative to* $\mathbf{d}$*.*

Background

How does computable categoricity relative to a degree behave?

Fact

A computable structure $\mathcal{A}$ is **relatively computably categorical** *if for all degrees* $\mathbf{d}$*,* $\mathcal{A}$ *is computably categorical relative to* $\mathbf{d}$*.*

We first begin with the following result.

Fact (Downey, Harrison-Trainor, Melnikov [DHTM21])

If $\mathcal{A}$ is a computable structure and it is computably categorical relative to some degree $\mathbf{d} \geq \mathbf{0}''$, then $\mathcal{A}$ has a $\mathbf{0}''$-computable Σ^0_1 Scott family. In particular, $\mathcal{A}$ is computably categorical relative to all $\mathbf{d} \geq \mathbf{0}''$.

The cone above $0''$

We sketch the proof of this fact. We first need the following results.

Theorem (Ash, Knight, Manasse, and Slaman [Ash+89]; Chisholm [Chi90])

A structure is relatively computably categorical if and only if it has a formally Σ_1 Scott family.

The cone above $\mathbf{0}''$

We sketch the proof of this fact. We first need the following results.

Theorem (Ash, Knight, Manasse, and Slaman [Ash+89]; Chisholm [Chi90])

A structure is relatively computably categorical if and only if it has a formally Σ_1 Scott family.

Theorem (Gončarov [Gon80])

If a structure is computably categorical and its $\forall\exists$ theory is decidable, then it is relatively computably categorical.

The cone above $\mathbf{0}''$

We sketch the proof of this fact. We first need the following results.

Theorem (Ash, Knight, Manasse, and Slaman [Ash+89]; Chisholm [Chi90])

A structure is relatively computably categorical if and only if it has a formally Σ_1 Scott family.

Theorem (Gončarov [Gon80])

If a structure is computably categorical and its $\forall\exists$ theory is decidable, then it is relatively computably categorical.

We'll use a relativized version of Gončarov's theorem in the proof sketch.

The cone above $\mathbf{0}''$

We sketch the proof of this fact. We first need the following results.

Theorem (Ash, Knight, Manasse, and Slaman [Ash+89]; Chisholm [Chi90])

A structure is relatively computably categorical if and only if it has a formally Σ_1 Scott family.

Theorem (Gončarov [Gon80] (relativized))

If a structure is computably categorical relative to $\mathbf{d}$ and its $\forall\exists$ theory is $\mathbf{d}$-decidable, then it has a Scott family of $\exists$-formulas that is c.e. in $\mathbf{d}$.

We'll use a relativized version of Gončarov's theorem in the proof sketch.

The cone above $\mathbf{0}''$: proof sketch

Proof sketch.

(1) Suppose $\mathcal{A}$ is computably categorical relative to a degree $\mathbf{d} \geq \mathbf{0}''$. Since $\mathcal{A}$ is computable, its $\forall\exists$ diagram is computable from $\mathbf{0}''$ and hence from $\mathbf{d}$.

The cone above $\mathbf{0}''$: proof sketch

Proof sketch.

(1) Suppose $\mathcal{A}$ is computably categorical relative to a degree $\mathbf{d} \geq \mathbf{0}''$. Since $\mathcal{A}$ is computable, its $\forall\exists$ diagram is computable from $\mathbf{0}''$ and hence from $\mathbf{d}$.

(2) By the relativized version of Gončarov's result, $\mathcal{A}$ has a formally Σ_1 Scott family c.e. in $\mathbf{d}$.

The cone above $\mathbf{0}''$: proof sketch

Proof sketch.

(1) Suppose $\mathcal{A}$ is computably categorical relative to a degree $\mathbf{d} \geq \mathbf{0}''$. Since $\mathcal{A}$ is computable, its $\forall\exists$ diagram is computable from $\mathbf{0}''$ and hence from $\mathbf{d}$.

(2) By the relativized version of Gončarov's result, $\mathcal{A}$ has a formally Σ_1 Scott family c.e. in $\mathbf{d}$.

(3) We can use $\mathbf{0}''$ to enumerate this Scott family of $\exists$-formulas, and so this is a formally Σ_1 Scott family relative to $\mathbf{0}''$.

The cone above $0''$: proof sketch

Proof sketch.

(1) Suppose $\mathcal{A}$ is computably categorical relative to a degree $\mathbf{d} \geq \mathbf{0}''$. Since $\mathcal{A}$ is computable, its $\forall\exists$ diagram is computable from $\mathbf{0}''$ and hence from $\mathbf{d}$.

(2) By the relativized version of Gončarov's result, $\mathcal{A}$ has a formally Σ_1 Scott family c.e. in $\mathbf{d}$.

(3) We can use $\mathbf{0}''$ to enumerate this Scott family of $\exists$-formulas, and so this is a formally Σ_1 Scott family relative to $\mathbf{0}''$.

Using this Scott family, we can computably build isomorphisms, and so for every $\mathbf{d} \geq \mathbf{0}''$, $\mathcal{A}$ is computably categorical relative to $\mathbf{d}$.

The cone above $0''$: proof sketch

Proof sketch.

(1) Suppose $\mathcal{A}$ is computably categorical relative to a degree $\mathbf{d} \geq \mathbf{0}''$. Since $\mathcal{A}$ is computable, its $\forall\exists$ diagram is computable from $\mathbf{0}''$ and hence from $\mathbf{d}$.

(2) By the relativized version of Gončarov's result, $\mathcal{A}$ has a formally Σ_1 Scott family c.e. in $\mathbf{d}$.

(3) We can use $\mathbf{0}''$ to enumerate this Scott family of $\exists$-formulas, and so this is a formally Σ_1 Scott family relative to $\mathbf{0}''$.

Using this Scott family, we can computably build isomorphisms, and so for every $\mathbf{d} \geq \mathbf{0}''$, $\mathcal{A}$ is computably categorical relative to $\mathbf{d}$. This fact implies that for *any* computable structure $\mathcal{A}$, either it is computably categorical relative to *all* degrees above $\mathbf{0}''$ or to *no* degree above $\mathbf{0}''$.

Below $\mathbf{0}'$

In the c.e. degrees, being computably categorical relative to a degree is *not* monotonic.

Below $\mathbf{0}'$

In the c.e. degrees, being computably categorical relative to a degree is *not* monotonic.

Theorem (Downey, Harrison-Trainor, Melnikov [DHTM21])

There is a computable structure $\mathcal{A}$ *and c.e. degrees* $\mathbf{0} = \mathbf{d}_0 <_T \mathbf{e}_0 <_T \mathbf{d}_1 <_T \mathbf{e}_1 <_T \dots$ *such that*

Below $\mathbf{0}'$

In the c.e. degrees, being computably categorical relative to a degree is *not* monotonic.

Theorem (Downey, Harrison-Trainor, Melnikov [DHTM21])

There is a computable structure $\mathcal{A}$ and c.e. degrees $\mathbf{0} = \mathbf{d}_0 <_T \mathbf{e}_0 <_T \mathbf{d}_1 <_T \mathbf{e}_1 <_T \dots$ such that

(1) *$\mathcal{A}$ is computably categorical relative to $\mathbf{d}_i$ for each i,*

(2) *$\mathcal{A}$ is not computably categorical relative to $\mathbf{e}_i$ for each i,*

(3) *$\mathcal{A}$ is computably categorical relative to $\mathbf{0}'$.*

Below $\mathbf{0}'$

In the c.e. degrees, being computably categorical relative to a degree is *not* monotonic.

Theorem (Downey, Harrison-Trainor, Melnikov [DHTM21])

There is a computable structure $\mathcal{A}$ and c.e. degrees $\mathbf{0} = \mathbf{d}_0 <_T \mathbf{e}_0 <_T \mathbf{d}_1 <_T \mathbf{e}_1 <_T \dots$ such that

(1) *$\mathcal{A}$ is computably categorical relative to $\mathbf{d}_i$ for each i,*

(2) *$\mathcal{A}$ is not computably categorical relative to $\mathbf{e}_i$ for each i,*

(3) *$\mathcal{A}$ is computably categorical relative to $\mathbf{0}'$.*

The structure they constructed to witness this was a directed graph.

Generalizing to partial orders of c.e. degrees

We generalize this result to partial orders of c.e. degrees.

Generalizing to partial orders of c.e. degrees

We generalize this result to partial orders of c.e. degrees.

Theorem (V. [Vil24])

Let $P = (P, \leq)$ be a computable partially ordered set and let $P = P_0 \sqcup P_1$ be a computable partition. Then, there exists a computable directed graph $\mathcal{G}$ and an embedding h of P into the c.e. degrees where

(1) *$\mathcal{G}$ is computably categorical;*

(2) *$\mathcal{G}$ is computably categorical relative to each degree in $h(P_0)$; and*

(3) *$\mathcal{G}$ is not computably categorical relative to each degree in $h(P_1)$.*

Extensions of current work

Future directions: embedding a lattice

The techniques utilized in proving the poset result can also be combined with the usual techniques to construct minimal pairs.

Theorem (V. [Vil24])

There exists a computable computably categorical directed graph $\mathcal{G}$ and c.e. sets X_0 and X_1 such that

(1) *X_0 and X_1 form a minimal pair,*

(2) *$\mathcal{G}$ is not computably categorical relative to X_0 and to X_1, and*

(3) *$\mathcal{G}$ is computably categorical relative to $X_0 \oplus X_1$.*

Future directions: embedding a lattice

The techniques utilized in proving the poset result can also be combined with the usual techniques to construct minimal pairs.

Theorem (V. [Vil24])

There exists a computable computably categorical directed graph $\mathcal{G}$ and c.e. sets X_0 and X_1 such that

(1) *X_0 and X_1 form a minimal pair,*
(2) *$\mathcal{G}$ is not computably categorical relative to X_0 and to X_1, and*
(3) *$\mathcal{G}$ is computably categorical relative to $X_0 \oplus X_1$.*

Question

Can you embed bigger distributive lattices into the c.e. degrees in a manner similar to the poset result?

Future directions: in the generic degrees

Definition

A degree **d** is **low for isomorphism** if for every pair of computable structures $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$.

Future directions: in the generic degrees

Definition

A degree **d** is **low for isomorphism** if for every pair of computable structures $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$.

Theorem (Franklin, Solomon [FS14])

Every 2-generic degree is low for isomorphism.

Future directions: in the generic degrees

Definition

A degree $\mathbf{d}$ is **low for isomorphism** if for every pair of computable structures $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\Delta^0_1} \mathcal{B}$.

Theorem (Franklin, Solomon [FS14])

Every 2-generic degree is low for isomorphism.

This means that there *cannot* be a computable structure $\mathcal{A}$ which is not computably categorical but is computably categorical relative to $\mathbf{d}$ for a 2-generic degree $\mathbf{d}$.

Future directions: in the generic degrees

Definition

A degree $\mathbf{d}$ is **low for isomorphism** if for every pair of computable structures $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} \cong_{\mathbf{d}} \mathcal{B}$ if and only if $\mathcal{A} \cong_{\Delta_1^0} \mathcal{B}$.

Theorem (Franklin, Solomon [FS14])

Every 2-generic degree is low for isomorphism.

This means that there *cannot* be a computable structure $\mathcal{A}$ which is not computably categorical but is computably categorical relative to $\mathbf{d}$ for a 2-generic degree $\mathbf{d}$.

Theorem

There exists a (properly) 1-generic G such that there is a computable directed graph $\mathcal{A}$ where $\mathcal{A}$ is not computably categorical but is computably categorical relative to G.

Future directions: identifying pathological behavior in classes of structures

Question

For structures other than directed graphs, can you produce an example which witnesses the pathological behavior in the poset result?

Future directions: identifying pathological behavior in classes of structures

Question

For structures other than directed graphs, can you produce an example which witnesses the pathological behavior in the poset result?

There are some results in the literature that give a negative result for certain classes of structures already.

Future directions: identifying pathological behavior in classes of structures

Question

For structures other than directed graphs, can you produce an example which witnesses the pathological behavior in the poset result?

There are some results in the literature that give a negative result for certain classes of structures already.

Theorem (Bazhenov [Baz14])

For every degree $\mathbf{d} < \mathbf{0}'$*, a computable Boolean algebra is* $\mathbf{d}$*-computably categorical if and only if it is computably categorical.*

Future directions: identifying pathological behavior in classes of structures

Corollary (from results in [Hir+02] and [Mil+18])

For the following classes of structures, there exists a computable example in each class which witnesses the behavior in the poset result:

(1) *symmetric, irreflexive graphs; partial orderings; lattices; rings with zero-divisors; integral domains of arbitrary characteristic; commutative semigroups; and 2-step nilpotent groups (by Theorem 1.22 of [Hir+02])*

(2) *countable fields (by Theorem 1.8 of [Mil+18])*

Future directions: identifying pathological behavior in classes of structures

Corollary (from results in [Hir+02] and [Mil+18])

For the following classes of structures, there exists a computable example in each class which witnesses the behavior in the poset result:

(1) *symmetric, irreflexive graphs; partial orderings; lattices; rings with zero-divisors; integral domains of arbitrary characteristic; commutative semigroups; and* 2*-step nilpotent groups (by Theorem* 1.22 *of [Hir+02])*

(2) *countable fields (by Theorem* 1.8 *of [Mil+18])*

Currently, the full picture is yet to be determined for some classes of structures, such as linear orderings.

Proof sketch of the poset result

Notation

For $p \in P$, we build uniformly c.e. sets A_p.

Definition

For $p \in P$, we define the c.e. set

$$D_p = \bigoplus_{q \leq p} A_q.$$

Our embedding will be the map $h(p) = D_p$.

We also have the following notation for convenience.

Definition

$$\overline{D_p} := \bigoplus_{q \neq p} A_q.$$

Notation

We use the following notation for graphs.

Notation

We use the following notation for graphs.

Definition

- $\mathcal{M}_e$ is the eth (partial) computable graph with domain ω where $E(x, y) \iff \Phi_e(x, y) = 1$ and $\neg E(x, y) \iff \Phi_e(x, y) = 0$.
- $\mathcal{M}_i^{D_p}$ is the ith (partial) D_p-computable graph with domain ω where $E(x, y) \iff \Phi_i^{D_p}(x, y) = 1$ and $\neg E(x, y) \iff \Phi_i^{D_p}(x, y) = 0$.

Requirements

We have the following requirements:

- $N_e^p : \Phi_e^{\overline{D_p}} \neq A_p$,
- S_e : if $\mathcal{G} \cong \mathcal{M}_e$, then there exists a computable isomorphism $f_e : \mathcal{G} \to \mathcal{M}_e$,
- for $p \in P_0$, T_i^p : if $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then there exists a D_p-computable isomorphism $g_i^{D_p} : \mathcal{G} \to \mathcal{M}_i^{D_p}$, and
- for $q \in P_1$, R_e^q : $\Phi_e^{D_q} : \mathcal{G} \to \mathcal{B}_q$ is not an isomorphism where $\mathcal{B}_q$ is a D_q-computable copy of $\mathcal{G}$ we build.

Requirements

We have the following requirements:

- $N_e^p : \Phi_e^{\overline{D_p}} \neq A_p$,
- S_e : if $\mathcal{G} \cong \mathcal{M}_e$, then there exists a computable isomorphism $f_e : \mathcal{G} \to \mathcal{M}_e$,
- for $p \in P_0$, T_i^p : if $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then there exists a D_p-computable isomorphism $g_i^{D_p} : \mathcal{G} \to \mathcal{M}_i^{D_p}$, and
- for $q \in P_1$, R_e^q : $\Phi_e^{D_q} : \mathcal{G} \to \mathcal{B}_q$ is not an isomorphism where $\mathcal{B}_q$ is a D_q-computable copy of $\mathcal{G}$ we build.

The N_e^p requirements ensure that h is an embedding of P into the c.e. degrees.

Requirements

We have the following requirements:

- $N_e^p : \Phi_e^{\overline{D_p}} \neq A_p$,
- S_e : if $\mathcal{G} \cong \mathcal{M}_e$, then there exists a computable isomorphism $f_e : \mathcal{G} \to \mathcal{M}_e$,
- for $p \in P_0$, T_i^p : if $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then there exists a D_p-computable isomorphism $g_i^{D_p} : \mathcal{G} \to \mathcal{M}_i^{D_p}$, and
- for $q \in P_1$, R_e^q : $\Phi_e^{D_q} : \mathcal{G} \to \mathcal{B}_q$ is not an isomorphism where $\mathcal{B}_q$ is a D_q-computable copy of $\mathcal{G}$ we build.

The N_e^p requirements ensure that h is an embedding of P into the c.e. degrees. The S_e requirements ensure that $\mathcal{G}$ is computably categorical.

Requirements

We have the following requirements:

- N_e^p : $\Phi_e^{\overline{D_p}} \neq A_p$,
- S_e : if $\mathcal{G} \cong \mathcal{M}_e$, then there exists a computable isomorphism $f_e : \mathcal{G} \to \mathcal{M}_e$,
- for $p \in P_0$, T_i^p : if $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then there exists a D_p-computable isomorphism $g_i^{D_p} : \mathcal{G} \to \mathcal{M}_i^{D_p}$, and
- for $q \in P_1$, R_e^q : $\Phi_e^{D_q} : \mathcal{G} \to \mathcal{B}_q$ is not an isomorphism where $\mathcal{B}_q$ is a D_q-computable copy of $\mathcal{G}$ we build.

The N_e^p requirements ensure that h is an embedding of P into the c.e. degrees. The S_e requirements ensure that $\mathcal{G}$ is computably categorical. The T_i^p requirements ensure that $\mathcal{G}$ is computably categorical relative to all degrees in $h(P_0)$.

Requirements

We have the following requirements:

- $N_e^p : \Phi_e^{\overline{D_p}} \neq A_p$,
- S_e : if $\mathcal{G} \cong \mathcal{M}_e$, then there exists a computable isomorphism $f_e : \mathcal{G} \to \mathcal{M}_e$,
- for $p \in P_0$, T_i^p : if $\mathcal{G} \cong \mathcal{M}_i^{D_p}$, then there exists a D_p-computable isomorphism $g_i^{D_p} : \mathcal{G} \to \mathcal{M}_i^{D_p}$, and
- for $q \in P_1$, R_e^q : $\Phi_e^{D_q} : \mathcal{G} \to \mathcal{B}_q$ is not an isomorphism where $\mathcal{B}_q$ is a D_q-computable copy of $\mathcal{G}$ we build.

The N_e^p requirements ensure that h is an embedding of P into the c.e. degrees. The S_e requirements ensure that $\mathcal{G}$ is computably categorical. The T_i^p requirements ensure that $\mathcal{G}$ is computably categorical relative to all degrees in $h(P_0)$. The R_e^q requirements ensure that $\mathcal{G}$ is not computably categorical relative to any degree in $h(P_1)$.

Building $\mathcal{G}$ in stages

We build the computable directed graph $\mathcal{G}$ in stages.

Building $\mathcal{G}$ in stages

We build the computable directed graph $\mathcal{G}$ in stages.

At stage $s = 0$, we set the domain of $\mathcal{G}$ to be empty.

Building $\mathcal{G}$ in stages

We build the computable directed graph $\mathcal{G}$ in stages.

At stage $s = 0$, we set the domain of $\mathcal{G}$ to be empty.

At stage $s > 0$, we add two new connected components by adding a_{2s} and a_{2s+1} as root nodes. We attach 2-loop to each node. Then, we attach a $(5s+1)$-loop to a_{2s} and a $(5s+2)$-loop to a_{2s+1}.

Building $\mathcal{G}$ in stages

We build the computable directed graph $\mathcal{G}$ in stages.

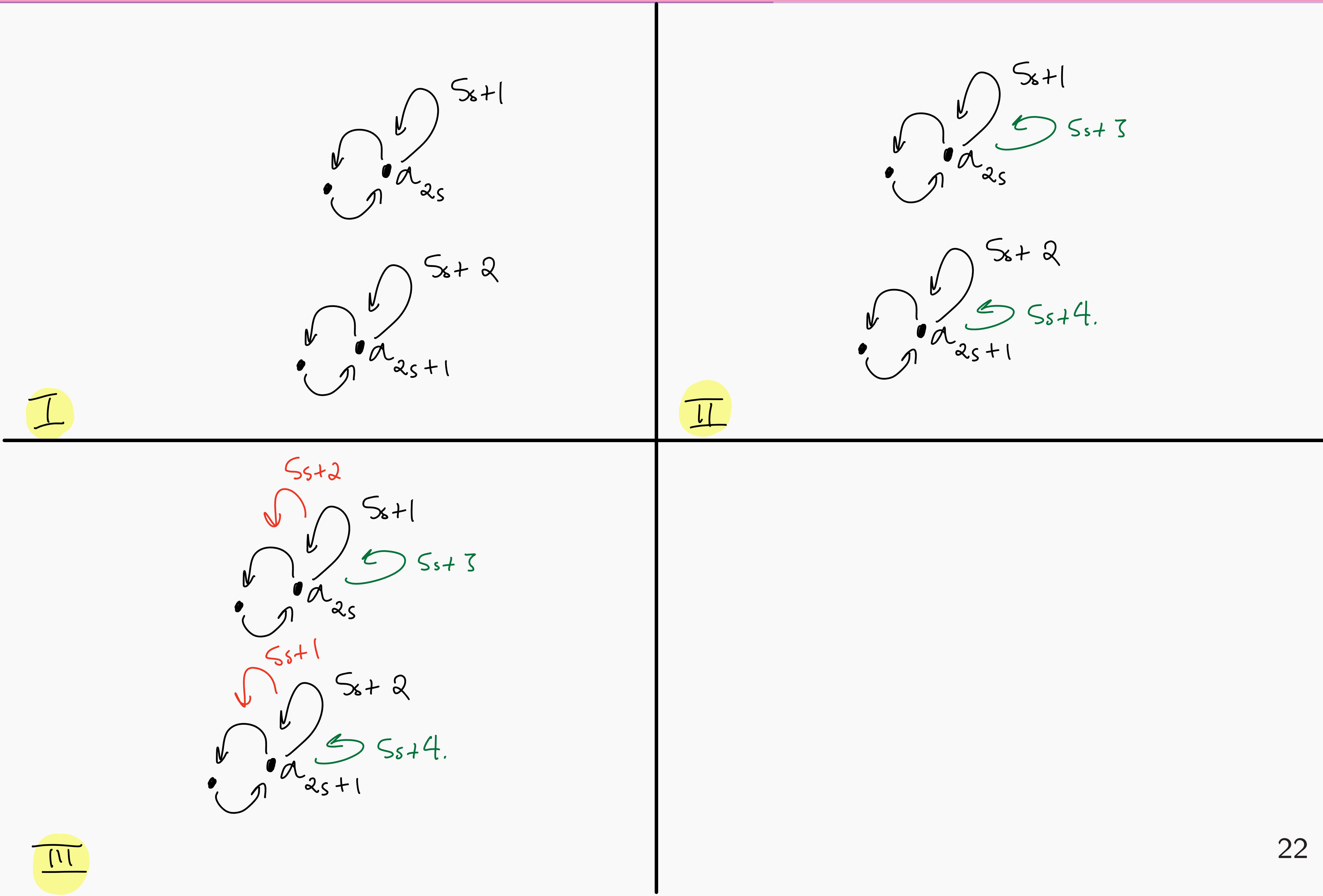
At stage $s = 0$, we set the domain of $\mathcal{G}$ to be empty.

At stage $s > 0$, we add two new connected components by adding a_{2s} and a_{2s+1} as root nodes. We attach 2-loop to each node. Then, we attach a $(5s + 1)$-loop to a_{2s} and a $(5s + 2)$-loop to a_{2s+1}.

Definition

The root node a_{2s} in our graph $\mathcal{G}$ with its loops is the 2s**th connected component** or just the 2sth component of $\mathcal{G}$.

Configuration of loops in $\mathcal{G}$



Basic strategies: N_e^p

This is our basic strategy to satisfy all N_e^p for $p \in P$.

Basic strategies: N_e^p

This is our basic strategy to satisfy all N_e^p for $p \in P$.

Let s be the current stage of the construction and let α be an N_e^p-strategy.

1. If α is first eligible to act at stage s, it defines its witness x_α to be a large unused number.

Basic strategies: N_e^p

This is our basic strategy to satisfy all N_e^p for $p \in P$.

Let s be the current stage of the construction and let α be an N_e^p-strategy.

1. If α is first eligible to act at stage s, it defines its witness x_α to be a large unused number.
2. Check if $\Phi_e^{\overline{D_p}}(x_\alpha)[s] \downarrow = 0$ and keep x_α out of A_p.

Basic strategies: N_e^p

This is our basic strategy to satisfy all N_e^p for $p \in P$.

Let s be the current stage of the construction and let α be an N_e^p-strategy.

1. If α is first eligible to act at stage s, it defines its witness x_α to be a large unused number.
2. Check if $\Phi_e^{\overline{D_p}}(x_\alpha)[s]\downarrow = 0$ and keep x_α out of A_p. If not, α takes no action at stage s.

Basic strategies: N_e^p

This is our basic strategy to satisfy all N_e^p for $p \in P$.

Let s be the current stage of the construction and let α be an N_e^p-strategy.

1. If α is first eligible to act at stage s, it defines its witness x_α to be a large unused number.
2. Check if $\Phi_e^{\overline{D_p}}(x_\alpha)[s]\downarrow= 0$ and keep x_α out of A_p. If not, α takes no action at stage s. If so, α enumerates x_α into A_p and restrains $A_p \upharpoonright (\text{use}(\Phi_e^{\overline{D_p}}(x_\alpha)) + 1)$.

Basic strategies: S_e

This is our basic strategy to satisfy all S_e requirements to make $\mathcal{G}$ computably categorical.

Basic strategies: S_e

This is our basic strategy to satisfy all S_e requirements to make $\mathcal{G}$ computably categorical.

Let s be the current stage of the construction and let α be an S_e-strategy.

1. If α is first eligible to act at stage s, it sets its parameter $n_\alpha = 0$. It looks for copies in $\mathcal{M}_e[s]$ of the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{G}[s]$. It defines $f_\alpha[s]$ to be the empty map.

Basic strategies: S_e

This is our basic strategy to satisfy all S_e requirements to make $\mathcal{G}$ computably categorical.

Let s be the current stage of the construction and let α be an S_e-strategy.

1. If α is first eligible to act at stage s, it sets its parameter $n_\alpha = 0$. It looks for copies in $\mathcal{M}_e[s]$ of the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{G}[s]$. It defines $f_\alpha[s]$ to be the empty map.
2. If n_α is defined and $f_\alpha[s-1]$ is defined for all $m < n_\alpha$, α looks for copies of the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{G}[s]$.

Basic strategies: S_e

3. If no copies of the $2n_\alpha$th and $(2n_\alpha + 1)$st components are found, α takes no additional action at stage s, retains the value of n_α, and sets $f_\alpha[s] = f_\alpha[s-1]$.

Basic strategies: S_e

3. If no copies of the $2n_\alpha$th and $(2n_\alpha + 1)$st components are found, α takes no additional action at stage s, retains the value of n_α, and sets $f_\alpha[s] = f_\alpha[s-1]$. If copies are found, α extends $f_\alpha[s-1]$ to $f_\alpha[s]$ by matching the components in $\mathcal{G}[s]$ to the copies found in $\mathcal{M}_e[s]$ and increments n_α by 1.

Basic strategies: T_i^p

Let $p \in P_0$. Our basic strategy to satisfy all T_i^p requirements to make $\mathcal{G}$ computably categorical relative to D_p is similar to our S_e-strategy. Let α be a T_i^p-strategy.

Basic strategies: T_i^p

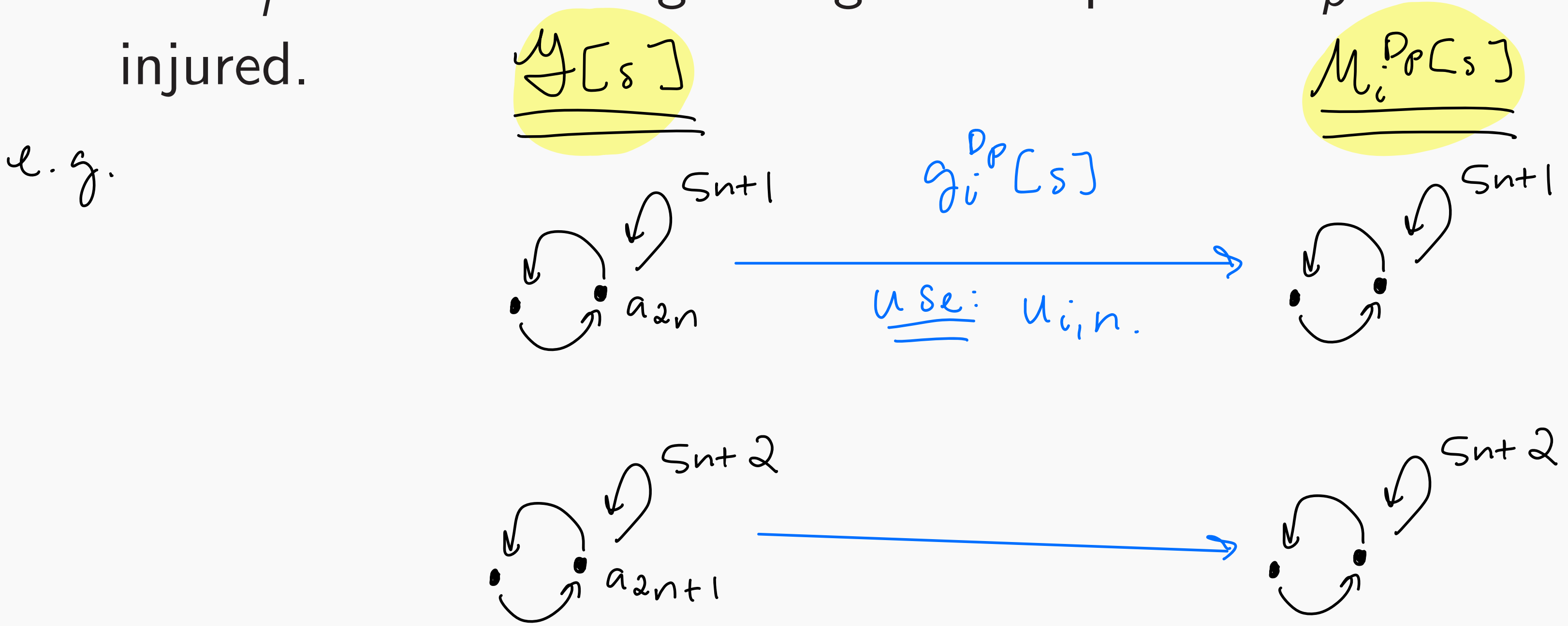
Let $p \in P_0$. Our basic strategy to satisfy all T_i^p requirements to make $\mathcal{G}$ computably categorical relative to D_p is similar to our S_e-strategy. Let α be a T_i^p-strategy.

For each n, we try to find copies of the $2n$th and $(2n+1)$st components of $\mathcal{G}$ in $\mathcal{M}_i^{D_p}$.

Basic strategies: T_i^p

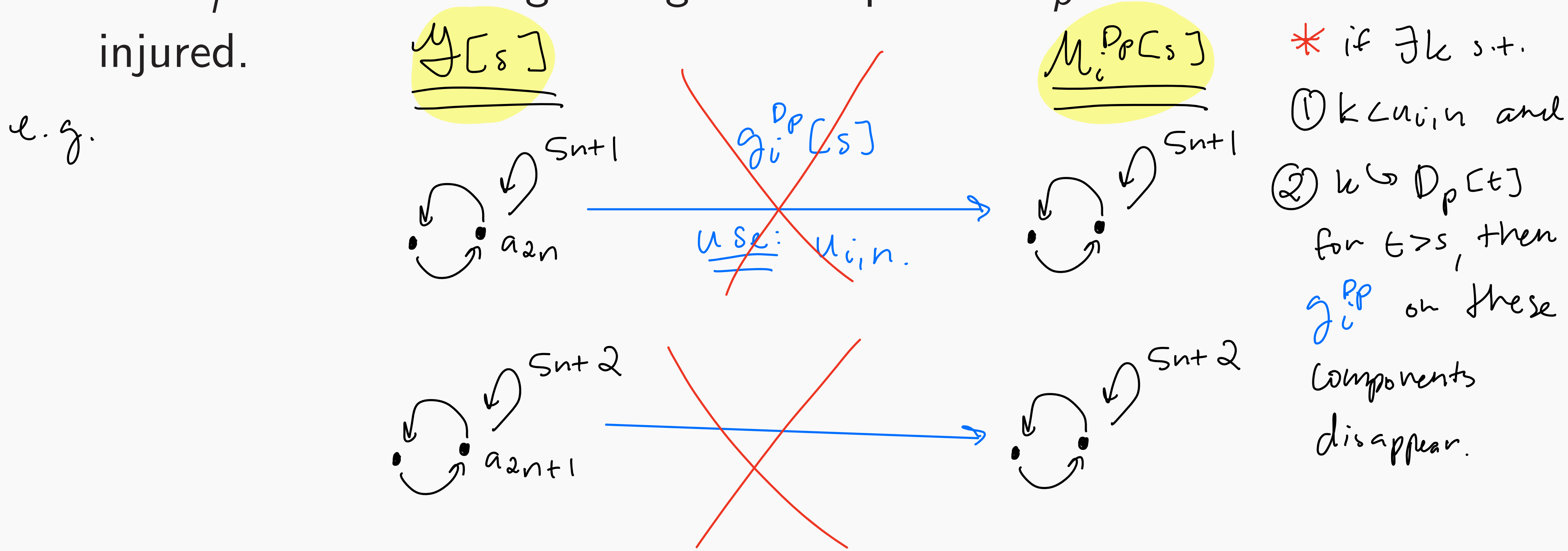
Let $p \in P_0$. Our basic strategy to satisfy all T_i^p requirements to make $\mathcal{G}$ computably categorical relative to D_p is similar to our S_e-strategy. Let α be a T_i^p-strategy.

For each n, we try to find copies of the $2n$th and $(2n+1)$st components of $\mathcal{G}$ in $\mathcal{M}_i^{D_p}$. But now because D_p is a c.e. set, loops in $\mathcal{M}_i^{D_p}$ or embeddings using a finite part of D_p as an oracle be injured.



Let $p \in P_0$. Our basic strategy to satisfy all T_i^p requirements to make $\mathcal{G}$ computably categorical relative to D_p is similar to our S_e-strategy. Let α be a T_i^p-strategy.

For each n, we try to find copies of the 2nth and $(2n+1)$st components of $\mathcal{G}$ in $\mathcal{M}_i^{D_p}$. But now because D_p is a c.e. set, loops in $\mathcal{M}_i^{D_p}$ or embeddings using a finite part of D_p as an oracle be injured.



Basic strategies: T_i^p

When α is next eligible to act at stage s, it will check if $D_p[t] \neq D_p[s]$ where t is the previous α-stage.

Basic strategies: T_i^p

When α is next eligible to act at stage s, it will check if $D_p[t] \neq D_p[s]$ where t is the previous α-stage.

If $D_p[t] \neq D_p[s]$, then α will update its parameter n_α accordingly depending on what type of injury occurred.

Basic strategies: T_i^p

When α is next eligible to act at stage s, it will check if $D_p[t] \neq D_p[s]$ where t is the previous α-stage.

If $D_p[t] \neq D_p[s]$, then α will update its parameter n_α accordingly depending on what type of injury occurred. Otherwise, it will proceed to try and match the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{G}$ for the n_α parameter it had at the beginning of stage s.

Basic strategies: R_e^q

Finally, for $q \in P_1$, we do the following to satisfy all R_e^q requirements to make $\mathcal{G}$ not computably categorical relative to D_q.

Basic strategies: R_e^q

Finally, for $q \in P_1$, we do the following to satisfy all R_e^q requirements to make $\mathcal{G}$ not computably categorical relative to D_q.

We will build a D_q-computable graph $\mathcal{B}_q$ which is isomorphic to $\mathcal{G}$ in stages, similarly to how we built $\mathcal{G}$.

Basic strategies: R_e^q

Finally, for $q \in P_1$, we do the following to satisfy all R_e^q requirements to make $\mathcal{G}$ not computably categorical relative to D_q.

We will build a D_q-computable graph $\mathcal{B}_q$ which is isomorphic to $\mathcal{G}$ in stages, similarly to how we built $\mathcal{G}$. At stage $s = 0$, let $\mathcal{B}_q = \emptyset$.

Basic strategies: R_e^q

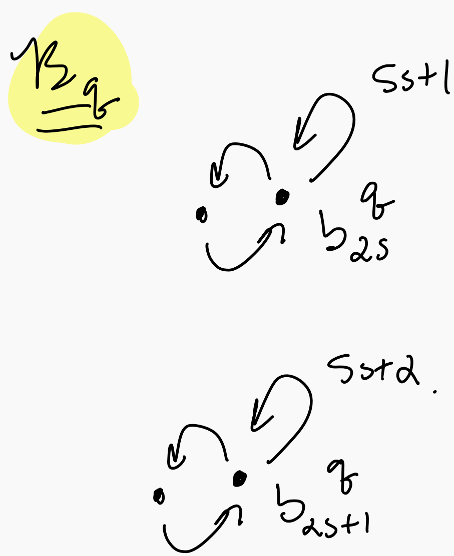
Finally, for $q \in P_1$, we do the following to satisfy all R_e^q requirements to make $\mathcal{G}$ not computably categorical relative to D_q.

We will build a D_q-computable graph $\mathcal{B}_q$ which is isomorphic to $\mathcal{G}$ in stages, similarly to how we built $\mathcal{G}$. At stage $s = 0$, let $\mathcal{B}_q = \emptyset$. At stage $s > 0$, add two new root nodes b_{2s}^q and b_{2s+1}^q and attach to each one a 2-loop.

Basic strategies: R_e^q

Finally, for $q \in P_1$, we do the following to satisfy all R_e^q requirements to make $\mathcal{G}$ not computably categorical relative to D_q.

We will build a D_q-computable graph $\mathcal{B}_q$ which is isomorphic to $\mathcal{G}$ in stages, similarly to how we built $\mathcal{G}$. At stage $s = 0$, let $\mathcal{B}_q = \emptyset$. At stage $s > 0$, add two new root nodes b_{2s}^q and b_{2s+1}^q and attach to each one a 2-loop. Attach a $(5s+1)$-loop to b_{2s}^q and a $(5s+2)$-loop to b_{2s+1}^q.



Basic strategies: R_e^q

This is our diagonalization strategy to satisfy all R_e^q.

Basic strategies: R_e^q

This is our diagonalization strategy to satisfy all R_e^q.

Let s be the current stage of the construction and let α be an R_e^q-strategy.

1. If α is first eligible to act at stage s, it defines its parameter n_α to be a large unused number.

Basic strategies: R_e^q

This is our diagonalization strategy to satisfy all R_e^q.

Let s be the current stage of the construction and let α be an R_e^q-strategy.

1. If α is first eligible to act at stage s, it defines its parameter n_α to be a large unused number.
2. α checks if $\Phi_e^{D_q}[s]$ maps the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{G}[s]$ to the corresponding copies in $\mathcal{B}_q[s]$.

Basic strategies: R_e^q

This is our diagonalization strategy to satisfy all R_e^q.

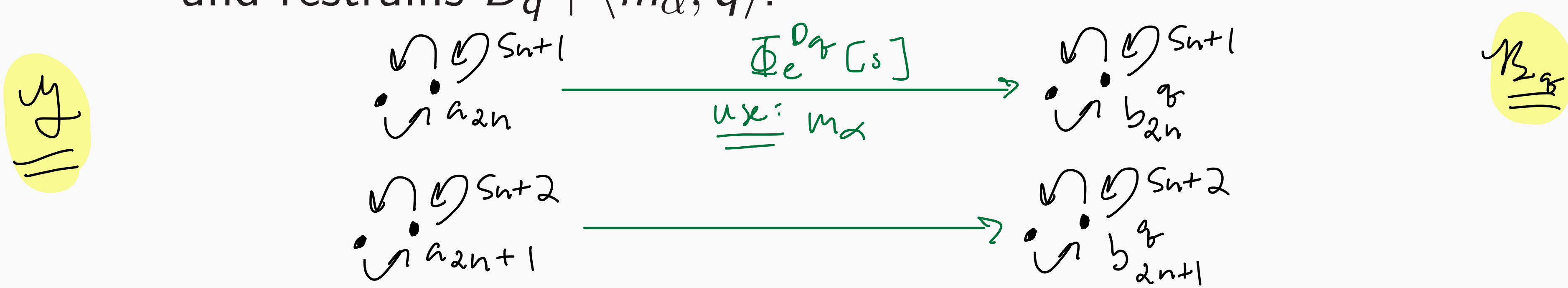
Let s be the current stage of the construction and let α be an R_e^q-strategy.

1. If α is first eligible to act at stage s, it defines its parameter n_α to be a large unused number.
2. α checks if $\Phi_e^{D_q}[s]$ maps the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{G}[s]$ to the corresponding copies in $\mathcal{B}_q[s]$. If not, α takes no further action.

This is our diagonalization strategy to satisfy all R_e^q.

Let s be the current stage of the construction and let α be an R_e^q-strategy.

1. If α is first eligible to act at stage s, it defines its parameter n_α to be a large unused number.
2. α checks if $\Phi_e^{D_q}[s]$ maps the $2n_\alpha$th and $(2n_\alpha + 1)$st components of $\mathcal{G}[s]$ to the corresponding copies in $\mathcal{B}_q[s]$. If not, α takes no further action. If α sees such a computation, it defines m_α to be the max of the uses of these computations and restrains $D_q \upharpoonright \langle m_\alpha, q \rangle$.



Basic strategies: R_e^q

3. α attaches a $(5n+3)$-loop to a_{2n} and b^q_{2n} and a $(5n+4)$-loop to a_{2n+1} and b^q_{2n+1}.

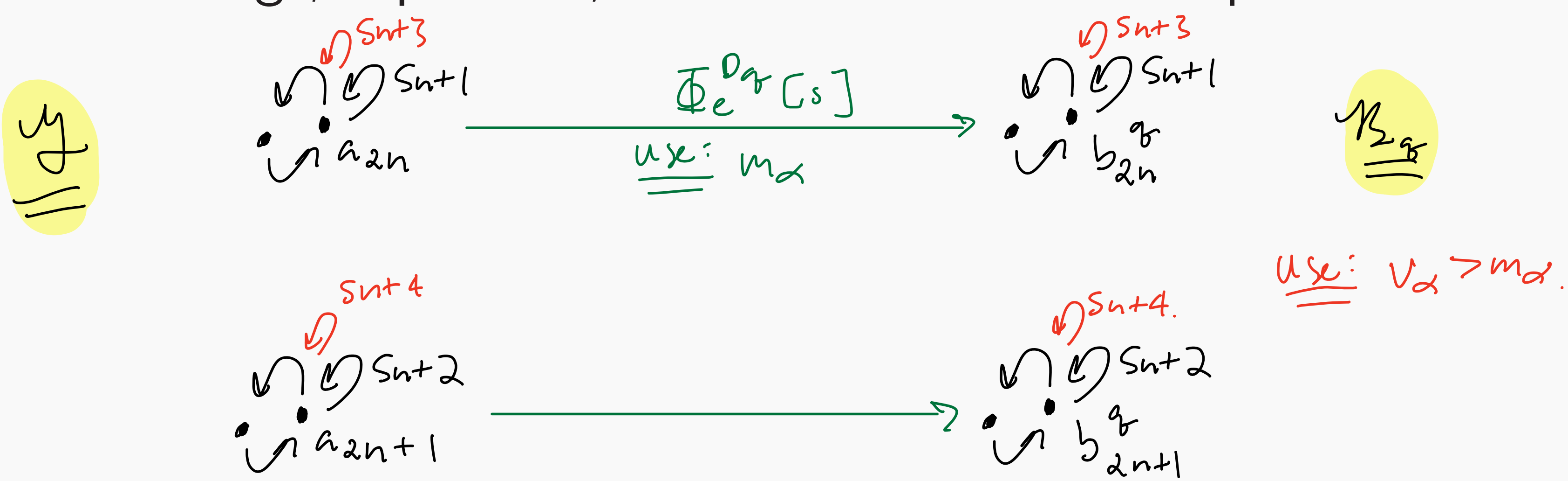
Basic strategies: R_e^q

3. α attaches a $(5n+3)$-loop to a_{2n} and b^q_{2n} and a $(5n+4)$-loop to a_{2n+1} and b^q_{2n+1}. Let v_α be the use associated with these loops appearing in $\mathcal{B}_q$. Note that $v_\alpha > m_\alpha$.

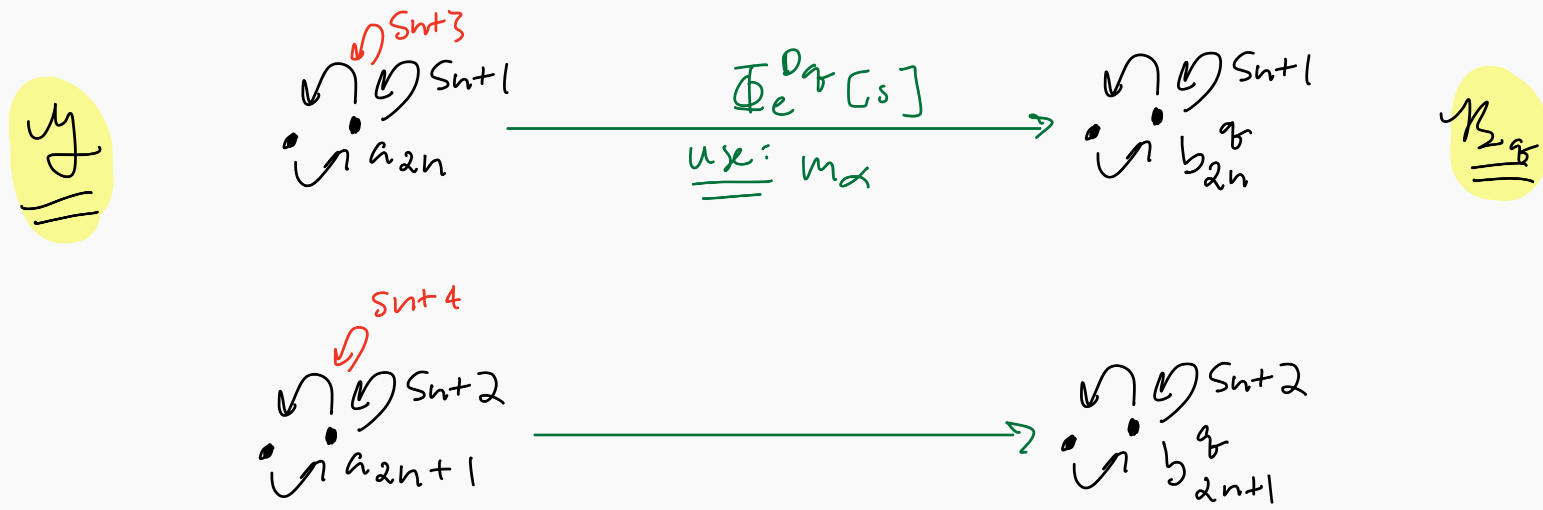
Basic strategies: R_e^q

3. α attaches a $(5n+3)$-loop to a_{2n} and b_{2n}^q and a $(5n+4)$-loop to a_{2n+1} and b_{2n+1}^q. Let v_α be the use associated with these loops appearing in $\mathcal{B}_q$. Note that $v_\alpha > m_\alpha$.
4. α now issues a challenge to all higher priority requirements which are S_e and T_i^p:

3. α attaches a $(5n+3)$-loop to a_{2n} and b^q_{2n} and a $(5n+4)$-loop to a_{2n+1} and b^q_{2n+1}. Let v_α be the use associated with these loops appearing in $\mathcal{B}_q$. Note that $v_\alpha > m_\alpha$.
4. α now issues a challenge to all higher priority requirements which are S_e and T_i^p: they must now extend their embeddings, if possible, to include these new loops.



5. If all higher S_e and T_i^p requirements can meet this challenge and α becomes eligible to act again at a later stage, it enumerates v_α into A_q. This makes the $(5n+3)$- and $(5n+4)$-loops in $\mathcal{B}_q$ disappear.

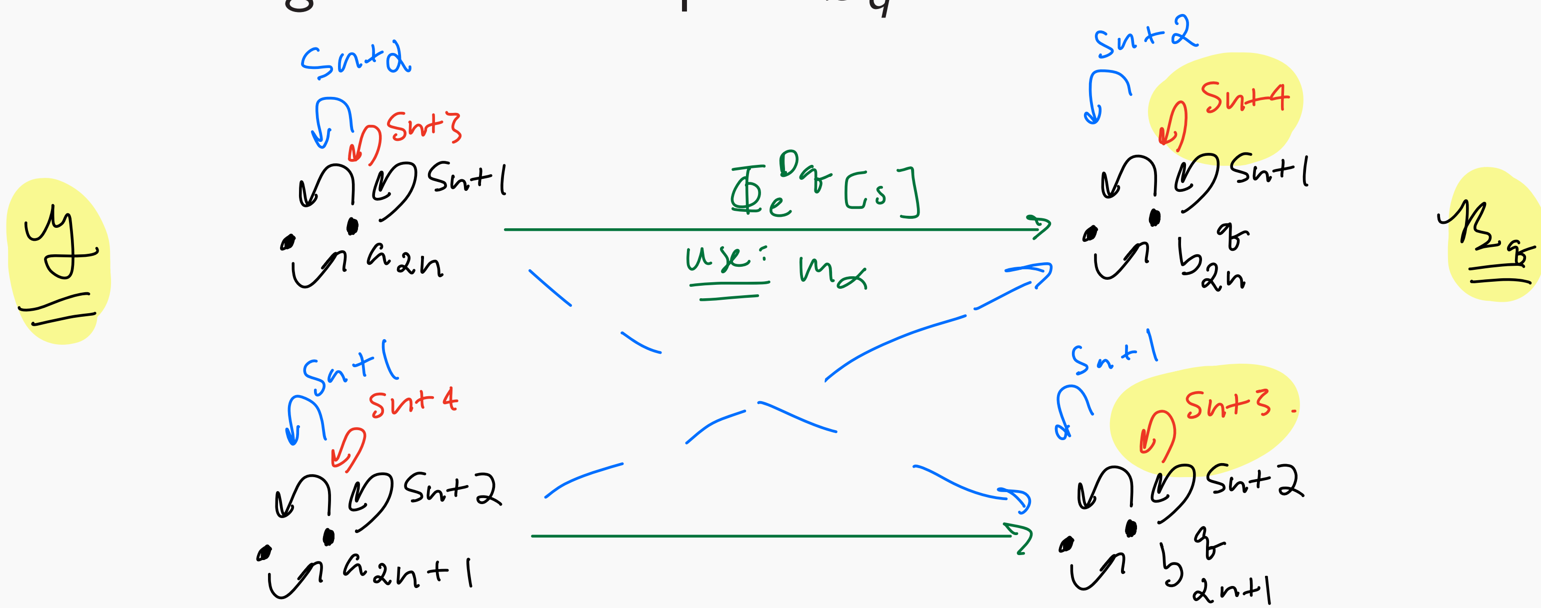


Basic strategies: R_e^q

6. α reattaches a $(5n+3)$-loop to b^q_{2n+1} and a $(5n+4)$-loop to b^q_{2n}. It also attaches a $(5n+1)$-loop to a_{2n+1} and to b^q_{2n+1}, and a $(5n+2)$-loop to a_{2n} and to b^q_{2n}.

6. α reattaches a $(5n+3)$-loop to b^q_{2n+1} and a $(5n+4)$-loop to b^q_{2n}. It also attaches a $(5n+1)$-loop to a_{2n+1} and to b^q_{2n+1}, and a $(5n+2)$-loop to a_{2n} and to b^q_{2n}.

Our final configuration of loops in $\mathcal{B}_q$ is now:



Interactions between strategies

There are several interactions and conflicts to keep note of in the construction.

Interactions between strategies

There are several interactions and conflicts to keep note of in the construction.

Interaction 1

The R_e^q-strategy wants to diagonalize while the S_e and T_i^p-strategies want to build embeddings: this was resolved by having R_e^q "wait" for higher priority S_e and T_i^p requirements and the homogenizing part of step 6 in the R_e^q-strategy.

Interactions between strategies

There are several interactions and conflicts to keep note of in the construction.

Interaction 1

The R_e^q-strategy wants to diagonalize while the S_e and T_i^p-strategies want to build embeddings: this was resolved by having R_e^q "wait" for higher priority S_e and T_i^p requirements and the homogenizing part of step 6 in the R_e^q-strategy.

Interaction 2

The N_e^p-strategy must enumerate numbers into A_p to achieve independence of degrees: this is resolved on a tree of strategies and by letting T_i^p check for any changes in D_p up to a finite part each stage.

Interactions between strategies

The last important interaction comes from the poset ordering on P.

Interactions between strategies

The last important interaction comes from the poset ordering on P.

Interaction 3

An R_e^q-strategy β and a T_i^p-strategy α when $q < p$ in P and T_i^p is of higher priority than R_e^q: the T_i^p-strategy needs an additional step for when it is challenged to enumerate any uses associated to the $2n_\beta$th and $(2n_\beta + 1)$st components of $\mathcal{G}$ into A_p. This lets us lift uses for T_i^p so it can succeed.

Thanks

Thank you for your attention!

I'd be happy to answer any questions.

References

[Ash+89] C. Ash et al. **"Generic copies of countable structures".** *Annals of Pure and Applied Logic* 42.3 (1989), pp. 195–205. ISSN: 0168-0072.

[Baz14] N. A. Bazhenov. **"Δ^0_2-Categoricity of Boolean Algebras".** *Journal of Mathematical Sciences* 203.4 (2014), pp. 444–454.

[Chi90] J. Chisholm. **"Effective Model Theory vs. Recursive Model Theory".** *The Journal of Symbolic Logic* 55.3 (1990), pp. 1168–1191. ISSN: 00224812.

[DHTM21] R. Downey, M. Harrison-Trainor, and A. Melnikov. **"Relativizing computable categoricity"**. *Proc. Amer. Math. Soc.* 149.9 (2021), pp. 3999–4013. ISSN: 0002-9939.

[FS14] J. N. Y. Franklin and R. Solomon. **"Degrees that Are Low for Isomorphism"**. *Computability* 3 (2014), pp. 73–89.

[Gon77] S. S. Gončarov. **"The quantity of nonautoequivalent constructivizations"**. *Algebra and Logic* 16.3 (May 1977), pp. 169–185. ISSN: 1573-8302.

[Gon80] S. S. Gončarov. **"The problem of the number of nonautoequivalent constructivizations"**. *Algebra i Logika* 19.6 (1980), pp. 621–639, 745.

[Hir+02] D. R. Hirschfeldt et al. **"Degree spectra and computable dimensions in algebraic structures".** *Annals of Pure and Applied Logic* 115.1 (2002), pp. 71–113. ISSN: 0168-0072.

[Mil+18] R. Miller et al. **"A Computable Functor from Graphs to Fields".** *The Journal of Symbolic Logic* 83.1 (2018), 326–348.

[Vil24] J. D. Villano. **Computable categoricity relative to a c.e. degree.** 2024. arXiv: 2401.06641 [math.LO].