

Embeddings, \sim_α , and Abelian *P*-groups

S. VanDenDriessche

Department of Mathematics
University of Notre Dame

CiE, June 28, 2011

Classification

In many branches of math, we would like to classify structures with respect to some notion of equivalence by invariants.

- Given classes of structures, can we determine which has a more difficult classification problem?
- The structures usually form a Polish space, K , in which the equivalence relation, E is definable (in some sense).
- H. Friedman and L. Stanley exploited this to create a Borel reducibility for classes of countable structures.

Classification

In many branches of math, we would like to classify structures with respect to some notion of equivalence by invariants.

- Given classes of structures, can we determine which has a more difficult classification problem?
- The structures usually form a Polish space, K , in which the equivalence relation, E is definable (in some sense).
- H. Friedman and L. Stanley exploited this to create a Borel reducibility for classes of countable structures.

Classification

In many branches of math, we would like to classify structures with respect to some notion of equivalence by invariants.

- Given classes of structures, can we determine which has a more difficult classification problem?
- The structures usually form a Polish space, K , in which the equivalence relation, E is definable (in some sense).
- H. Friedman and L. Stanley exploited this to create a Borel reducibility for classes of countable structures.

Classification

In many branches of math, we would like to classify structures with respect to some notion of equivalence by invariants.

- Given classes of structures, can we determine which has a more difficult classification problem?
- The structures usually form a Polish space, K , in which the equivalence relation, E is definable (in some sense).
- H. Friedman and L. Stanley exploited this to create a Borel reducibility for classes of countable structures.

Turing Computable Embeddings

Knight, et al. considered only computable languages, and structures with universes subsets of ω , in order to formulate an effective analog. We further generalize their definition to allow classification for equivalence relations other than isomorphism.

Definition

A *Turing computable embedding* of (K, E) into (K', E') is an operator $\Phi = \phi_e$ such that

- for each $\mathcal{A} \in K$ there exists $\mathcal{B} \in K'$ such that $\Phi(\mathcal{A}) = \phi_e^{D(\mathcal{A})} = \chi_{D(\mathcal{B})}$, and
- if $\mathcal{A}, \mathcal{A}' \in K$, then $\mathcal{A}E\mathcal{A}' \leftrightarrow \Phi(\mathcal{A})E'\Phi(\mathcal{A}')$.

This induces a preordering, denoted by $(K, E) \leq_{tc} (K', E')$.

Turing Computable Embeddings

Knight, et al. considered only computable languages, and structures with universes subsets of ω , in order to formulate an effective analog. We further generalize their definition to allow classification for equivalence relations other than isomorphism.

Definition

A *Turing computable embedding* of (K, E) into (K', E') is an operator $\Phi = \phi_e$ such that

- for each $\mathcal{A} \in K$ there exists $\mathcal{B} \in K'$ such that $\Phi(\mathcal{A}) = \phi_e^{D(\mathcal{A})} = \chi_{D(\mathcal{B})}$, and
- if $\mathcal{A}, \mathcal{A}' \in K$, then $\mathcal{A}E\mathcal{A}' \leftrightarrow \Phi(\mathcal{A})E'\Phi(\mathcal{A}')$.

This induces a preordering, denoted by $(K, E) \leq_{tc} (K', E')$.

Turing Computable Embeddings

Knight, et al. considered only computable languages, and structures with universes subsets of ω , in order to formulate an effective analog. We further generalize their definition to allow classification for equivalence relations other than isomorphism.

Definition

A *Turing computable embedding* of (K, E) into (K', E') is an operator $\Phi = \phi_e$ such that

- for each $\mathcal{A} \in K$ there exists $\mathcal{B} \in K'$ such that $\Phi(\mathcal{A}) = \phi_e^{D(\mathcal{A})} = \chi_{D(\mathcal{B})}$, and
- if $\mathcal{A}, \mathcal{A}' \in K$, then $\mathcal{A}E\mathcal{A}' \leftrightarrow \Phi(\mathcal{A})E'\Phi(\mathcal{A}')$.

This induces a preordering, denoted by $(K, E) \leq_{tc} (K', E')$.

Turing Computable Embeddings

Knight, et al. considered only computable languages, and structures with universes subsets of ω , in order to formulate an effective analog. We further generalize their definition to allow classification for equivalence relations other than isomorphism.

Definition

A *Turing computable embedding* of (K, E) into (K', E') is an operator $\Phi = \phi_e$ such that

- for each $\mathcal{A} \in K$ there exists $\mathcal{B} \in K'$ such that $\Phi(\mathcal{A}) = \phi_e^{D(\mathcal{A})} = \chi_{D(\mathcal{B})}$, and
- if $\mathcal{A}, \mathcal{A}' \in K$, then $\mathcal{A}E\mathcal{A}' \leftrightarrow \Phi(\mathcal{A})E'\Phi(\mathcal{A}')$.

This induces a preordering, denoted by $(K, E) \leq_{tc} (K', E')$.

Some Results

- $PF <_{tc} FLO <_{tc} FVS <_{tc} VS <_{tc} LO$.
- For all K , $K \leq_{tc} UG \equiv_{tc} LO$.
- $VS \equiv_{tc} ACF \equiv_{tc} ZS$
- $K \leq_{tc} VS$ if and only if there is a computable sequence $(\phi_n)_{n \in \omega}$ of Σ_2^C sentences in the language of K such that
 - for $\mathcal{A} \in K$, and $m < n$, if $\mathcal{A} \models \phi_n$ then $\mathcal{A} \models \phi_m$ and
 - for $\mathcal{A}, \mathcal{B} \in K$, if $\mathcal{A} \not\cong \mathcal{B}$ then there is some n such that ϕ_n is true in only one of \mathcal{A}, \mathcal{B} .

Some Results

- $PF <_{tc} FLO <_{tc} FVS <_{tc} VS <_{tc} LO$.
- For all K , $K \leq_{tc} UG \equiv_{tc} LO$.
- $VS \equiv_{tc} ACF \equiv_{tc} ZS$
- $K \leq_{tc} VS$ if and only if there is a computable sequence $(\phi_n)_{n \in \omega}$ of Σ_2^c sentences in the language of K such that
 - for $\mathcal{A} \in K$, and $m < n$, if $\mathcal{A} \models \phi_n$ then $\mathcal{A} \models \phi_m$ and
 - for $\mathcal{A}, \mathcal{B} \in K$, if $\mathcal{A} \not\cong \mathcal{B}$ then there is some n such that ϕ_n is true in only one of \mathcal{A}, \mathcal{B} .

Some Results

- $PF <_{tc} FLO <_{tc} FVS <_{tc} VS <_{tc} LO$.
- For all K , $K \leq_{tc} UG \equiv_{tc} LO$.
- $VS \equiv_{tc} ACF \equiv_{tc} ZS$
- $K \leq_{tc} VS$ if and only if there is a computable sequence $(\phi_n)_{n \in \omega}$ of Σ_2^C sentences in the language of K such that
 - for $\mathcal{A} \in K$, and $m < n$, if $\mathcal{A} \models \phi_n$ then $\mathcal{A} \models \phi_m$ and
 - for $\mathcal{A}, \mathcal{B} \in K$, if $\mathcal{A} \not\cong \mathcal{B}$ then there is some n such that ϕ_n is true in only one of \mathcal{A}, \mathcal{B} .

Some Results

- $PF <_{tc} FLO <_{tc} FVS <_{tc} VS <_{tc} LO$.
- For all K , $K \leq_{tc} UG \equiv_{tc} LO$.
- $VS \equiv_{tc} ACF \equiv_{tc} ZS$
- $K \leq_{tc} VS$ if and only if there is a computable sequence $(\phi_n)_{n \in \omega}$ of Σ_2^c sentences in the language of K such that
 - for $\mathcal{A} \in K$, and $m < n$, if $\mathcal{A} \models \phi_n$ then $\mathcal{A} \models \phi_m$ and
 - for $\mathcal{A}, \mathcal{B} \in K$, if $\mathcal{A} \not\cong \mathcal{B}$ then there is some n such that ϕ_n is true in only one of \mathcal{A}, \mathcal{B} .

Computable Infinitary Sentences

The *computable infinitary formulas* are formulas of $\mathcal{L}_{\omega_1\omega}$ where all disjunction/conjunctions are computably enumerable.

Definition

- The Σ_0^c and Π_0^c formulas are the finitary quantifier-free formulas (we suppose they are always in normal form).
- For a computable ordinal $\alpha > 0$,
 - a Σ_α^c formula $\phi(\bar{x})$ is a c.e. disjunction of formulas of the form $(\exists \bar{u})\psi(\bar{x}, \bar{u})$, where each $\psi \in \Pi_\beta^c$ for $\beta < \alpha$.
 - a Π_α^c formula $\phi(\bar{x})$ is a c.e. conjunction of formulas of the form $(\forall \bar{u})\psi(\bar{x}, \bar{u})$, where each $\psi \in \Sigma_\beta^c$ for $\beta < \alpha$.

Note that $neg(\phi)$ is defined in the obvious way.

Computable Infinitary Sentences

The *computable infinitary formulas* are formulas of $\mathcal{L}_{\omega_1\omega}$ where all disjunction/conjunctions are computably enumerable.

Definition

- The Σ_0^c and Π_0^c formulas are the finitary quantifier-free formulas (we suppose they are always in normal form).
- For a computable ordinal $\alpha > 0$,
 - a Σ_α^c formula $\phi(\bar{x})$ is a c.e. disjunction of formulas of the form $(\exists \bar{u})\psi(\bar{x}, \bar{u})$, where each $\psi \in \Pi_\beta^c$ for $\beta < \alpha$.
 - a Π_α^c formula $\phi(\bar{x})$ is a c.e. conjunction of formulas of the form $(\forall \bar{u})\psi(\bar{x}, \bar{u})$, where each $\psi \in \Sigma_\beta^c$ for $\beta < \alpha$.

Note that $neg(\phi)$ is defined in the obvious way.

The Pull-back Theorem

A powerful tool for showing non-embeddability is:

Theorem (Pull-back Theorem (Knight, et al.))

If $(K, E) \leq_{tc} (K', E')$ via Φ , then for any computable infinitary sentence ϕ in the language of K' , we can (effectively) find a computable infinitary sentence ϕ^ in the language of K such that*

- *for all $\mathcal{A} \in K$, $\Phi(\mathcal{A}) \models \phi$ if and only if $\mathcal{A} \models \phi^*$*
- *ϕ and ϕ^* have the same complexity.*

Note that this also gives necessary conditions for embeddings to exist (i.e. VS result).

The Pull-back Theorem

A powerful tool for showing non-embeddability is:

Theorem (Pull-back Theorem (Knight, et al.))

If $(K, E) \leq_{tc} (K', E')$ via Φ , then for any computable infinitary sentence ϕ in the language of K' , we can (effectively) find a computable infinitary sentence ϕ^ in the language of K such that*

- *for all $\mathcal{A} \in K$, $\Phi(\mathcal{A}) \models \phi$ if and only if $\mathcal{A} \models \phi^*$*
- *ϕ and ϕ^* have the same complexity.*

Note that this also gives necessary conditions for embeddings to exist (i.e. VS result).

The Pull-back Theorem

A powerful tool for showing non-embeddability is:

Theorem (Pull-back Theorem (Knight, et al.))

If $(K, E) \leq_{tc} (K', E')$ via Φ , then for any computable infinitary sentence ϕ in the language of K' , we can (effectively) find a computable infinitary sentence ϕ^ in the language of K such that*

- *for all $\mathcal{A} \in K$, $\Phi(\mathcal{A}) \models \phi$ if and only if $\mathcal{A} \models \phi^*$*
- *ϕ and ϕ^* have the same complexity.*

Note that this also gives necessary conditions for embeddings to exist (i.e. VS result).

P-groups

Fix any prime, p . Recall that an *Abelian p -group* is an Abelian group where each element has order a power of p . Let \mathcal{G} be a countable Abelian p -group.

- Define inductively: $\mathcal{G}_0 = \mathcal{G}$, $\mathcal{G}_{\beta+1} = p\mathcal{G}_\beta$, and $\mathcal{G}_\lambda = \bigcap_{\gamma < \lambda} \mathcal{G}_\gamma$.
- \mathcal{G} is *divisible* if every $x \in \mathcal{G}$ is divisible by p^n for all n .
- For each \mathcal{G} , there is a length, λ such that $\mathcal{G}_\lambda = \mathcal{G}_{\lambda+1}$.
- If $\mathcal{G}_\lambda = \{0\}$, we call \mathcal{G} *reduced*.

P-groups

Fix any prime, p . Recall that an *Abelian p -group* is an Abelian group where each element has order a power of p . Let \mathcal{G} be a countable Abelian p -group.

- Define inductively: $\mathcal{G}_0 = \mathcal{G}$, $\mathcal{G}_{\beta+1} = p\mathcal{G}_\beta$, and $\mathcal{G}_\lambda = \bigcap_{\gamma < \lambda} \mathcal{G}_\gamma$.
- \mathcal{G} is *divisible* if every $x \in \mathcal{G}$ is divisible by p^n for all n .
- For each \mathcal{G} , there is a length, λ such that $\mathcal{G}_\lambda = \mathcal{G}_{\lambda+1}$.
- If $\mathcal{G}_\lambda = \{0\}$, we call \mathcal{G} *reduced*.

P-groups

Fix any prime, p . Recall that an *Abelian p-group* is an Abelian group where each element has order a power of p . Let \mathcal{G} be a countable Abelian p -group.

- Define inductively: $\mathcal{G}_0 = \mathcal{G}$, $\mathcal{G}_{\beta+1} = p\mathcal{G}_\beta$, and $\mathcal{G}_\lambda = \bigcap_{\gamma < \lambda} \mathcal{G}_\gamma$.
- \mathcal{G} is *divisible* if every $x \in \mathcal{G}$ is divisible by p^n for all n .
- For each \mathcal{G} , there is a length, λ such that $\mathcal{G}_\lambda = \mathcal{G}_{\lambda+1}$.
- If $\mathcal{G}_\lambda = \{0\}$, we call \mathcal{G} *reduced*.

P-groups

Fix any prime, p . Recall that an *Abelian p-group* is an Abelian group where each element has order a power of p . Let \mathcal{G} be a countable Abelian p -group.

- Define inductively: $\mathcal{G}_0 = \mathcal{G}$, $\mathcal{G}_{\beta+1} = p\mathcal{G}_\beta$, and $\mathcal{G}_\lambda = \bigcap_{\gamma < \lambda} \mathcal{G}_\gamma$.
- \mathcal{G} is *divisible* if every $x \in \mathcal{G}$ is divisible by p^n for all n .
- For each \mathcal{G} , there is a length, λ such that $\mathcal{G}_\lambda = \mathcal{G}_{\lambda+1}$.
- If $\mathcal{G}_\lambda = \{0\}$, we call \mathcal{G} *reduced*.

P-groups

Fix any prime, p . Recall that an *Abelian p-group* is an Abelian group where each element has order a power of p . Let \mathcal{G} be a countable Abelian p -group.

- Define inductively: $\mathcal{G}_0 = \mathcal{G}$, $\mathcal{G}_{\beta+1} = p\mathcal{G}_\beta$, and $\mathcal{G}_\lambda = \bigcap_{\gamma < \lambda} \mathcal{G}_\gamma$.
- \mathcal{G} is *divisible* if every $x \in \mathcal{G}$ is divisible by p^n for all n .
- For each \mathcal{G} , there is a length, λ such that $\mathcal{G}_\lambda = \mathcal{G}_{\lambda+1}$.
- If $\mathcal{G}_\lambda = \{0\}$, we call \mathcal{G} *reduced*.

P-groups

Fix any prime, p . Recall that an *Abelian p-group* is an Abelian group where each element has order a power of p . Let \mathcal{G} be a countable Abelian p -group.

- Define inductively: $\mathcal{G}_0 = \mathcal{G}$, $\mathcal{G}_{\beta+1} = p\mathcal{G}_\beta$, and $\mathcal{G}_\lambda = \bigcap_{\gamma < \lambda} \mathcal{G}_\gamma$.
- \mathcal{G} is *divisible* if every $x \in \mathcal{G}$ is divisible by p^n for all n .
- For each \mathcal{G} , there is a length, λ such that $\mathcal{G}_\lambda = \mathcal{G}_{\lambda+1}$.
- If $\mathcal{G}_\lambda = \{0\}$, we call \mathcal{G} *reduced*.

Invariants

- Let $P_\beta(\mathcal{G}) := \{x \in \mathcal{G}_\beta : px = 0\}$.
- Note that $P_\beta(\mathcal{G})/P_{\beta+1}(\mathcal{G})$ is a \mathbb{Z}_p -vector space, and let $u_\beta(\mathcal{G})$ be its dimension.

Theorem (Ulm)

Two countable, reduced Abelian p -groups are isomorphic if and only if they have the same Ulm invariants.

Invariants

- Let $P_\beta(\mathcal{G}) := \{x \in \mathcal{G}_\beta : px = 0\}$.
- Note that $P_\beta(\mathcal{G})/P_{\beta+1}(\mathcal{G})$ is a \mathbb{Z}_p -vector space, and let $u_\beta(\mathcal{G})$ be its dimension.

Theorem (Ulm)

Two countable, reduced Abelian p -groups are isomorphic if and only if they have the same Ulm invariants.

Invariants

- Let $P_\beta(\mathcal{G}) := \{x \in \mathcal{G}_\beta : px = 0\}$.
- Note that $P_\beta(\mathcal{G})/P_{\beta+1}(\mathcal{G})$ is a \mathbb{Z}_p -vector space, and let $u_\beta(\mathcal{G})$ be its dimension.

Theorem (Ulm)

Two countable, reduced Abelian p -groups are isomorphic if and only if they have the same Ulm invariants.

Khisamiev's Theorem

The following special case of a theorem of N. Khisamiev will prove useful.

Theorem (Khisamiev)

If G is a X'' -computable reduced Abelian p -group, then there is an X -computable reduced Abelian p -group H , such that

- 1 $H_\omega \cong G$
- 2 $U_n(H) = \infty$ for all $n \in \omega$
- 3 *Given an index for G , we can compute an index for H .*

Khisamiev's Theorem

The following special case of a theorem of N. Khisamiev will prove useful.

Theorem (Khisamiev)

If G is a X'' -computable reduced Abelian p -group, then there is an X -computable reduced Abelian p -group H , such that

- 1 $H_\omega \cong G$
- 2 $U_n(H) = \infty$ for all $n \in \omega$
- 3 *Given an index for G , we can compute an index for H .*

\sim_α

Motivated by the importance of Σ_α^c sentences in many examples, we define the following equivalence relation.

Definition

In any class K , satisfying our conventions, we define the equivalence relation

$$\mathcal{A} \sim_\alpha \mathcal{B} \leftrightarrow (\forall \phi \in \Sigma_\alpha^c)(\mathcal{A} \models \phi \leftrightarrow \mathcal{B} \models \phi).$$

Note that $\mathcal{A} \cong \mathcal{B} \rightarrow (\forall \alpha)\mathcal{A} \sim_\alpha \mathcal{B}$, but the reverse implication does not hold.

\sim_α

Motivated by the importance of Σ_α^c sentences in many examples, we define the following equivalence relation.

Definition

In any class K , satisfying our conventions, we define the equivalence relation

$$\mathcal{A} \sim_\alpha \mathcal{B} \leftrightarrow (\forall \phi \in \Sigma_\alpha^c)(\mathcal{A} \models \phi \leftrightarrow \mathcal{B} \models \phi).$$

Note that $\mathcal{A} \cong \mathcal{B} \rightarrow (\forall \alpha) \mathcal{A} \sim_\alpha \mathcal{B}$, but the reverse implication does not hold.

\sim_α

Motivated by the importance of Σ_α^c sentences in many examples, we define the following equivalence relation.

Definition

In any class K , satisfying our conventions, we define the equivalence relation

$$\mathcal{A} \sim_\alpha \mathcal{B} \leftrightarrow (\forall \phi \in \Sigma_\alpha^c)(\mathcal{A} \models \phi \leftrightarrow \mathcal{B} \models \phi).$$

Note that $\mathcal{A} \cong \mathcal{B} \rightarrow (\forall \alpha)\mathcal{A} \sim_\alpha \mathcal{B}$, but the reverse implication does not hold.

A Lemma

Let AB_α^p be the class of reduced Abelian p -groups of length α .

Lemma (V.)

For any class (K, E) , if $(K, E) \leq_{tc} AB_\omega^p$, then for any $A, B \in K$,

$$AEB \leftrightarrow A \sim_2 B.$$

Proof: Apply the Pull-back theorem to fact that members of AB_ω^p are distinguished by Σ_2^c sentences.

A Lemma

Let AB_α^p be the class of reduced Abelian p -groups of length α .

Lemma (V.)

For any class (K, E) , if $(K, E) \leq_{tc} AB_\omega^p$, then for any $A, B \in K$,

$$AEB \leftrightarrow A \sim_2 B.$$

Proof: Apply the Pull-back theorem to fact that members of AB_ω^p are distinguished by Σ_2^c sentences.

A Lemma

Let AB_α^p be the class of reduced Abelian p -groups of length α .

Lemma (V.)

For any class (K, E) , if $(K, E) \leq_{tc} AB_\omega^p$, then for any $A, B \in K$,

$$AEB \leftrightarrow A \sim_2 B.$$

Proof: Apply the Pull-back theorem to fact that members of AB_ω^p are distinguished by Σ_2^c sentences.

Length ω Case

Theorem (V.)

For any class K , $(K, \sim_2) \leq_{tc} (AB_\omega^p, \sim_2)$.

Proof: We modify proofs of S. Quinn to manually build $\Phi = \phi_e$.
First note that we can enumerate the Σ_2^c sentences:

$$\bigvee_{i \in A} (\exists \bar{u}_i) \bigwedge_{j \in B} (\forall \bar{v}_j) \psi_j(\bar{u}_i, \bar{v}_j).$$

Length ω Case

Theorem (V.)

For any class K , $(K, \sim_2) \leq_{tc} (AB_\omega^P, \sim_2)$.

Proof: We modify proofs of S. Quinn to manually build $\Phi = \phi_e$.
 First note that we can enumerate the Σ_2^c sentences:

$$\bigvee_{i \in A} (\exists \bar{u}_i) \bigwedge_{j \in B} (\forall \bar{v}_j) \psi_j(\bar{u}_i, \bar{v}_j).$$

Proof, cont.

- We exploit the Σ_2 guessing strategy to build a copy of

$$\mathbb{Z}_p^{m_1} \oplus \mathbb{Z}_{p^2}^{n_1} \oplus \mathbb{Z}_{p^3}^{m_2} \oplus \mathbb{Z}_{p^4}^{n_2} \oplus \dots,$$

where all $m_i \in \{0, 1\}$ and $n_j = \omega$ for all j .

- So long as we think a given $\phi_n \in \Sigma_2^c$ is true in the input structure, make $m_{n+1} = 1$ in the output group. If we find out we are wrong, reset it to 0 by trashing corresponding summand.
- It is easy to check that this is a Turing computable embedding.

Proof, cont.

- We exploit the Σ_2 guessing strategy to build a copy of

$$\mathbb{Z}_p^{m_1} \oplus \mathbb{Z}_{p^2}^{n_1} \oplus \mathbb{Z}_{p^3}^{m_2} \oplus \mathbb{Z}_{p^4}^{n_2} \oplus \dots,$$

where all $m_i \in \{0, 1\}$ and $n_j = \omega$ for all j .

- So long as we think a given $\phi_n \in \Sigma_2^c$ is true in the input structure, make $m_{n+1} = 1$ in the output group. If we find out we are wrong, reset it to 0 by trashing corresponding summand.
- It is easy to check that this is a Turing computable embedding.

Proof, cont.

- We exploit the Σ_2 guessing strategy to build a copy of

$$\mathbb{Z}_p^{m_1} \oplus \mathbb{Z}_{p^2}^{n_1} \oplus \mathbb{Z}_{p^3}^{m_2} \oplus \mathbb{Z}_{p^4}^{n_2} \oplus \dots,$$

where all $m_i \in \{0, 1\}$ and $n_j = \omega$ for all j .

- So long as we think a given $\phi_n \in \Sigma_2^c$ is true in the input structure, make $m_{n+1} = 1$ in the output group. If we find out we are wrong, reset it to 0 by trashing corresponding summand.
- It is easy to check that this is a Turing computable embedding.

Length $\omega \cdot m$ Case

Theorem (V.)

For any class K , $(K, \sim_n) \leq_{tc} (AB_{\omega \cdot m}^p, \sim_n)$ if and only if $n \leq 2m$.

Proof:

- Again, Pull-back theorem. For the other direction, we enumerate the relevant sentences, and use a guessing strategy (now using a \emptyset^{2m-2} oracle).
- We use the same guessing strategy to create (an index for) a group of length ω , but now we cannot computably output its diagram.
- We simultaneously lower the complexity and increase the length of the group by repeated application of Khisamiev's theorem, yielding a (index for a) computable group.

Length $\omega \cdot m$ Case

Theorem (V.)

For any class K , $(K, \sim_n) \leq_{tc} (AB_{\omega \cdot m}^P, \sim_n)$ if and only if $n \leq 2m$.

Proof:

- Again, Pull-back theorem. For the other direction, we enumerate the relevant sentences, and use a guessing strategy (now using a \emptyset^{2m-2} oracle).
- We use the same guessing strategy to create (an index for) a group of length ω , but now we cannot computably output its diagram.
- We simultaneously lower the complexity and increase the length of the group by repeated application of Khisamiev's theorem, yielding a (index for a) computable group.

Length $\omega \cdot m$ Case

Theorem (V.)

For any class K , $(K, \sim_n) \leq_{tc} (AB_{\omega \cdot m}^p, \sim_n)$ if and only if $n \leq 2m$.

Proof:

- Again, Pull-back theorem. For the other direction, we enumerate the relevant sentences, and use a guessing strategy (now using a \emptyset^{2m-2} oracle).
- We use the same guessing strategy to create (an index for) a group of length ω , but now we cannot computably output its diagram.
- We simultaneously lower the complexity and increase the length of the group by repeated application of Khisamiev's theorem, yielding a (index for a) computable group.

Length $\omega \cdot m$ Case

Theorem (V.)

For any class K , $(K, \sim_n) \leq_{tc} (AB_{\omega \cdot m}^p, \sim_n)$ if and only if $n \leq 2m$.

Proof:

- Again, Pull-back theorem. For the other direction, we enumerate the relevant sentences, and use a guessing strategy (now using a \emptyset^{2m-2} oracle).
- We use the same guessing strategy to create (an index for) a group of length ω , but now we cannot computably output its diagram.
- We simultaneously lower the complexity and increase the length of the group by repeated application of Khisamiev's theorem, yielding a (index for a) computable group.

Length ω^2

Theorem (V.)

For any class K , $(K, \sim_{\alpha}) \leq_{tc} (AB_{\omega^2}^p, \sim_{\alpha})$ if and only if $\alpha \leq \omega$.

Theorem (V.)

For any class K , $(K, \sim_{\alpha}) \leq_{tc} (AB_{\beta}^p, \sim_{\alpha})$ if and only if $\beta = \omega \cdot \gamma$ and $\alpha < \gamma$.

The proofs requires calculating the back and forth relations for $AB_{\omega^2}^p$ and larger lengths.

Length ω^2

Theorem (V.)

For any class K , $(K, \sim_{\alpha}) \leq_{tc} (AB_{\omega^2}^p, \sim_{\alpha})$ if and only if $\alpha \leq \omega$.

Theorem (V.)

For any class K , $(K, \sim_{\alpha}) \leq_{tc} (AB_{\beta}^p, \sim_{\alpha})$ if and only if $\beta = \omega \cdot \gamma$ and $\alpha < \gamma$.

The proofs requires calculating the back and forth relations for $AB_{\omega^2}^p$ and larger lengths.

Length ω^2

Theorem (V.)




For any class K , $(K, \sim_{\alpha}) \leq_{tc} (AB_{\omega^2}^p, \sim_{\alpha})$ if and only if $\alpha \leq \omega$.

Theorem (V.)

For any class K , $(K, \sim_{\alpha}) \leq_{tc} (AB_{\beta}^p, \sim_{\alpha})$ if and only if $\beta = \omega \cdot \gamma$ and $\alpha < \gamma$.

The proofs requires calculating the back and forth relations for $AB_{\omega^2}^p$ and larger lengths.

Bibliography I

-  W. Calvert, D. Cummins, J.F. Knight, and S. Miller,
“Comparing Classes of Finite Structures”
Algebra and Logic, 43(2004), pp.365-373, 2004.
-  Julia F. Knight, Sara Miller, and M. Vanden Boom,
“Turing Computable Embeddings”
Journal of Symbolic Logic, 72(2007), pp. 901-918, 2007.
-  Sara B. Quinn,
“Algorithmic Complexity of Algebraic Structures”
Doctoral Dissertation, 2008.