Computable structure theory and infinitary logic

Part 2: Infinitary logic and forcing

Matthew Harrison-Trainor
University of Illinois Chicago

ASL NA Meeting 2026 Tutorial Part 2

Today we explore a paradigm in computable structure theory which is espoused especially by Knight, but also by Montalbán, and others (including myself).

Today we explore a paradigm in computable structure theory which is espoused especially by Knight, but also by Montalbán, and others (including myself).

- We should try to understand how the computational properties of presentations of structures are related to structural properties of the abstract isomorphism type.

Today we explore a paradigm in computable structure theory which is espoused especially by Knight, but also by Montalbán, and others (including myself).

- We should try to understand how the computational properties of presentations of structures are related to structural properties of the abstract isomorphism type.
- These structural properties are often expressed in the infinitary logic $\mathcal{L}_{\omega_1\omega}$.

Today we explore a paradigm in computable structure theory which is espoused especially by Knight, but also by Montalbán, and others (including myself).

- We should try to understand how the computational properties of presentations of structures are related to structural properties of the abstract isomorphism type.
- These structural properties are often expressed in the infinitary logic $\mathcal{L}_{\omega_1\omega}$.
- Even when there is not a direct correspondence, we should still look through the lens of infinitary logic.

Today we explore a paradigm in computable structure theory which is espoused especially by Knight, but also by Montalbán, and others (including myself).

- We should try to understand how the computational properties of presentations of structures are related to structural properties of the abstract isomorphism type.
- These structural properties are often expressed in the infinitary logic $\mathcal{L}_{\omega_1\omega}$.
- Even when there is not a direct correspondence, we should still look through the lens of infinitary logic.

Let's start with two simple examples.

Consider $(\mathbb{N}, <)$ and the adjacency relation $\mathrm{Adj} \subseteq \mathbb{N}^2$.

Consider $(\mathbb{N}, <)$ and the adjacency relation $\mathrm{Adj} \subseteq \mathbb{N}^2$.

- Adj is computable.

Consider $(\mathbb{N}, <)$ and the adjacency relation $\mathrm{Adj} \subseteq \mathbb{N}^2$.

- Adj is computable.

In any copy $\mathcal{A} = (\mathbb{N}, \prec)$, we have $\mathrm{Adj}^{\mathcal{A}} \subseteq \mathbb{N}^2$.

Consider $(\mathbb{N}, <)$ and the adjacency relation $\mathrm{Adj} \subseteq \mathbb{N}^2$.

- Adj is computable.

In any copy $\mathcal{A} = (\mathbb{N}, \prec)$, we have $\mathrm{Adj}^{\mathcal{A}} \subseteq \mathbb{N}^2$.

- $\mathrm{Adj}^{\mathcal{A}}$ is co-c.e. relative to $\mathcal{A}$, i.e., we can $\mathcal{A}$-computably list the non-adjacent pairs.

Consider $(\mathbb{N}, <)$ and the adjacency relation $\mathrm{Adj} \subseteq \mathbb{N}^2$.

- Adj is computable.

In any copy $\mathcal{A} = (\mathbb{N}, \prec)$, we have $\mathrm{Adj}^{\mathcal{A}} \subseteq \mathbb{N}^2$.

- $\mathrm{Adj}^{\mathcal{A}}$ is co-c.e. relative to $\mathcal{A}$, i.e., we can $\mathcal{A}$-computably list the non-adjacent pairs.

Why?

Consider $(\mathbb{N}, <)$ and the adjacency relation $\mathrm{Adj} \subseteq \mathbb{N}^2$.

- Adj is computable.

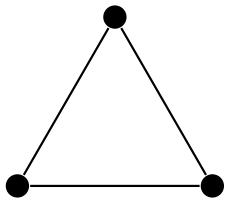
In any copy $\mathcal{A} = (\mathbb{N}, \prec)$, we have $\mathrm{Adj}^{\mathcal{A}} \subseteq \mathbb{N}^2$.

- $\mathrm{Adj}^{\mathcal{A}}$ is co-c.e. relative to $\mathcal{A}$, i.e., we can $\mathcal{A}$-computably list the non-adjacent pairs.

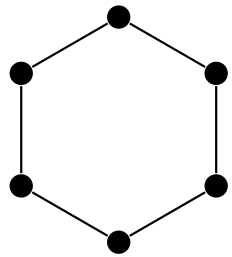
Why? Because Adj is definable by a universal formula:

$$\mathrm{Adj}(x, y) \iff x \neq y \ \wedge \ \forall z \, (z \geq x \wedge z \geq y) \vee (z \leq x \wedge z \leq y).$$

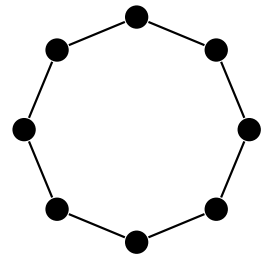
Let $X \subseteq \mathbb{N}$, and let $\mathcal{A}_X$ the graph with cycles of size $n+3$ for $n \in X$.



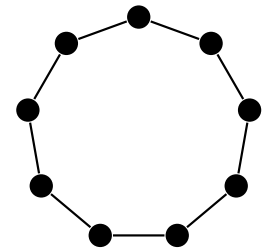
$0 \in X$



$3 \in X$

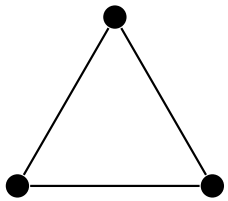


$5 \in X$

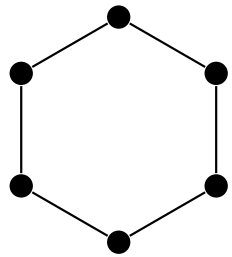


$6 \in X$

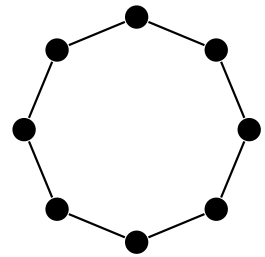
Let $X \subseteq \mathbb{N}$, and let $\mathcal{A}_X$ the graph with cycles of size $n+3$ for $n \in X$.



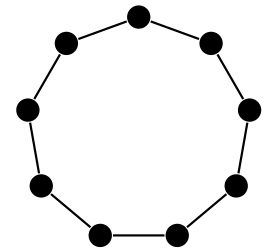
$0 \in X$



$3 \in X$



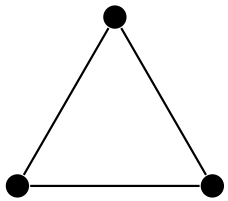
$5 \in X$



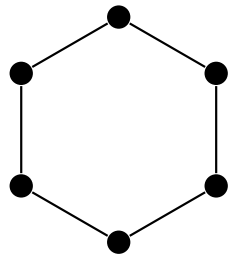
$6 \in X$

For every $\mathcal{B} \cong \mathcal{A}_X$, $\mathcal{B}$ can computably enumerate X.

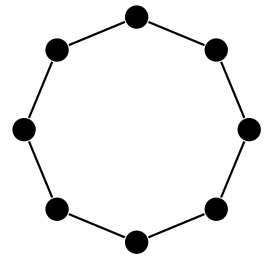
Let $X \subseteq \mathbb{N}$, and let $\mathcal{A}_X$ the graph with cycles of size $n+3$ for $n \in X$.



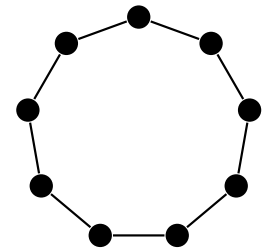
$0 \in X$



$3 \in X$



$5 \in X$



$6 \in X$

For every $\mathcal{B} \cong \mathcal{A}_X$, $\mathcal{B}$ can computably enumerate X.

This is because there is a computable list of existential formulas φ_n such that

$$n \in X \iff \mathcal{A}_X \models \varphi_n.$$

Let $\mathcal{A}$ be a structure. Suppose that there were computable lists of existential formulas $\varphi_{n,m}(\bar{x})$ and a tuple $\bar{a}$ such that

$$n \in X \iff \mathcal{A} \models \bigvee_m \varphi_{n,m}(\bar{a}).$$

Let $\mathcal{A}$ be a structure. Suppose that there were computable lists of existential formulas $\varphi_{n,m}(\bar{x})$ and a tuple $\bar{a}$ such that

$$n \in X \iff \mathcal{A} \models \bigvee_m \varphi_{n,m}(\bar{a}).$$

Then X would be c.e. relative to every copy of $\mathcal{A}$.

Let $\mathcal{A}$ be a structure. Suppose that there were computable lists of existential formulas $\varphi_{n,m}(\bar{x})$ and a tuple $\bar{a}$ such that

$$n \in X \iff \mathcal{A} \models \bigvee_m \varphi_{n,m}(\bar{a}).$$

Then X would be c.e. relative to every copy of $\mathcal{A}$.

Thus: We should use infinitary logic.

Definition

The logic $\mathcal{L}_{\omega_1\omega}$ is like standard first-order logic except that we add countably infinite conjunctions $\bigwedge$ and disjunctions $\bigvee$.

Definition

The logic $\mathcal{L}_{\omega_1\omega}$ is like standard first-order logic except that we add countably infinite conjunctions $\bigwedge$ and disjunctions $\bigvee$.

Example

An abelian group being torsion can be expressed by:

$$\forall x \bigvee_{n\in\mathbb{N}} \underbrace{x + \cdots + x}_{n \text{ times}} = 0.$$

Definition

The logic $\mathcal{L}_{\omega_1\omega}$ is like standard first-order logic except that we add countably infinite conjunctions $\bigwedge$ and disjunctions $\bigvee$.

Example

An abelian group being torsion can be expressed by:

$$\forall x \bigvee_{n\in\mathbb{N}} \underbrace{x+\cdots+x}_{n \text{ times}} = 0.$$

Example

A $\mathbb{Q}$-vector space having infinite dimension can be expressed by:

$$\bigwedge_{n\geq 1} \exists x_1,\ldots,x_n \bigwedge_{\substack{(c_1,\ldots,c_n)\in\mathbb{Q}^n \\ \bar{c}\neq\bar{0}}} c_1x_1+\cdots+c_nx_n \neq 0.$$

We measure the complexity of $\mathcal{L}_{\omega_1\omega}$ formulas by counting alternations of quantifiers. An infinitary disjunction $\bigvee$ counts as an existential quantifier, and an infinitary conjunction $\bigwedge$ as a universal quantifier.

We measure the complexity of $\mathcal{L}_{\omega_1\omega}$ formulas by counting alternations of quantifiers. An infinitary disjunction $\bigvee$ counts as an existential quantifier, and an infinitary conjunction $\bigwedge$ as a universal quantifier.

- φ is Π_0 and Σ_0 if it is finitary quantifier-free,

We measure the complexity of $\mathcal{L}_{\omega_1\omega}$ formulas by counting alternations of quantifiers. An infinitary disjunction $\bigvee$ counts as an existential quantifier, and an infinitary conjunction $\bigwedge$ as a universal quantifier.

- φ is Π_0 and Σ_0 if it is finitary quantifier-free,
- φ is Σ_α if it is of the form

$$\varphi = \bigvee_{i \in I} \exists \bar{x}_i \, \psi_i(\bar{x}_i)$$

where each ψ_i is Π_β for some $\beta < \alpha$,

We measure the complexity of $\mathcal{L}_{\omega_1\omega}$ formulas by counting alternations of quantifiers. An infinitary disjunction $\bigvee$ counts as an existential quantifier, and an infinitary conjunction $\bigwedge$ as a universal quantifier.

- φ is Π_0 and Σ_0 if it is finitary quantifier-free,
- φ is Σ_α if it is of the form

$$\varphi = \bigvee_{i \in I} \exists \bar{x}_i \, \psi_i(\bar{x}_i)$$

  where each ψ_i is Π_β for some $\beta < \alpha$,
- φ is Π_α if it is of the form

$$\varphi = \bigwedge_{i \in I} \forall \bar{x}_i \, \psi_i(\bar{x}_i)$$

  where each ψ_i is Σ_β for some $\beta < \alpha$.

Example

The sentence expressing that an abelian group is torsion,

$$\forall x \bigvee_{n \in \mathbb{N}} \underbrace{x + \cdots + x}_{n \text{ times}} = 0,$$

is Π_2.

Example

The sentence expressing that an abelian group is torsion,

$$\forall x \bigvee_{n\in\mathbb{N}} \underbrace{x+\cdots+x}_{n \text{ times}} = 0,$$

is Π_2.

Example

The sentence expressing that a $\mathbb{Q}$-vector space has infinite dimension,

$$\bigwedge_{n\in\mathbb{N}} \exists x_1,\ldots,x_n \bigwedge_{\substack{(c_1,\ldots,c_n)\in\mathbb{Q}^n \\ \bar{c}\neq\bar{0}}} c_1x_1+\cdots+c_nx_n \neq 0,$$

is Π_3.

Infinitary logic

For simplicity we will only consider finite α.

Infinitary logic

For simplicity we will only consider finite α.

A formula $\varphi(\bar{x})$ is computable if all of the conjunctions and disjunctions are taken over computable lists of formulas.

Infinitary logic

For simplicity we will only consider finite α.

A formula $\varphi(\bar{x})$ is computable if all of the conjunctions and disjunctions are taken over computable lists of formulas.

Example

The sentence expressing that an abelian group is torsion is computable: the disjuncts $nx = 0$ can be listed computably.

Infinitary logic

For simplicity we will only consider finite α.

A formula $\varphi(\bar{x})$ is computable if all of the conjunctions and disjunctions are taken over computable lists of formulas.

Example

The sentence expressing that an abelian group is torsion is computable: the disjuncts $nx = 0$ can be listed computably.

Example

Let $X \subseteq \mathbb{N}$ be a set which is not c.e. Then the formula

$$\bigvee_{n \in X} \underbrace{x + \cdots + x}_{n \text{ times}} = 0$$

expressing that the order of x is in X is not computable.

The arithmetic hierarchy

We defined the Halting problem as

$$K = 0' = \{e : \varphi_e(e) \text{ halts}\}.$$

The arithmetic hierarchy

We defined the Halting problem as

$$K = 0' = \{e : \varphi_e(e) \text{ halts}\}.$$

We can also relativize this to an oracle A, giving

$$A' = \{e : \Phi_e^A(e) \text{ halts}\}.$$

The arithmetic hierarchy

We defined the Halting problem as

$$K = 0' = \{e : \varphi_e(e) \text{ halts}\}.$$

We can also relativize this to an oracle A, giving

$$A' = \{e : \Phi_e^A(e) \text{ halts}\}.$$

Iterating, we get

$$A, A', A'', A''' = A^{(3)}, A'''' = A^{(4)}, \ldots.$$

This gives a strictly increasing function on Turing degrees.

The arithmetic hierarchy

We defined the Halting problem as

$$K = 0' = \{e : \varphi_e(e) \text{ halts}\}.$$

We can also relativize this to an oracle A, giving

$$A' = \{e : \Phi_e^A(e) \text{ halts}\}.$$

Iterating, we get

$$A, A', A'', A''' = A^{(3)}, A'''' = A^{(4)}, \ldots.$$

This gives a strictly increasing function on Turing degrees.

Definition

A set A is arithmetic if $A \leq_T 0^{(n)}$ for some n if A is definable in $(\mathbb{N}, 0, 1, +, -, \cdot)$.

The arithmetic hierarchy

Definition

A set $A \subseteq \mathbb{N}$ is Σ^0_n if it is, equivalently

- c.e. relative to $0^{(n-1)}$

The arithmetic hierarchy

Definition

A set $A \subseteq \mathbb{N}$ is Σ^0_n if it is, equivalently

- c.e. relative to $0^{(n-1)}$
- of the form $\{x : \exists y_1 \forall y_2 \cdots R(x; y_1, \ldots, y_n)\}$ with R computable

The arithmetic hierarchy

Definition

A set $A \subseteq \mathbb{N}$ is Σ^0_n if it is, equivalently

- c.e. relative to $0^{(n-1)}$
- of the form $\{x : \exists y_1 \forall y_2 \cdots R(x; y_1, \ldots, y_n)\}$ with R computable
- definable by a Σ^0_n formula in $(\mathbb{N}, 0, 1, +, \cdot)$

The arithmetic hierarchy

Definition

A set $A \subseteq \mathbb{N}$ is Σ^0_n if it is, equivalently

- c.e. relative to $0^{(n-1)}$
- of the form $\{x : \exists y_1 \forall y_2 \cdots R(x; y_1, \ldots, y_n)\}$ with R computable
- definable by a Σ^0_n formula in $(\mathbb{N}, 0, 1, +, \cdot)$

A set $A \subseteq \mathbb{N}$ is Π^0_n if its complement is Σ^0_n.

The arithmetic hierarchy

Definition

A set $A \subseteq \mathbb{N}$ is Σ^0_n if it is, equivalently

- c.e. relative to $0^{(n-1)}$
- of the form $\{x : \exists y_1 \forall y_2 \cdots R(x; y_1, \ldots, y_n)\}$ with R computable
- definable by a Σ^0_n formula in $(\mathbb{N}, 0, 1, +, \cdot)$

A set $A \subseteq \mathbb{N}$ is Π^0_n if its complement is Σ^0_n.

This can all be relativized to an oracle, writing "Σ^0_n relative to A".

Theorem (Knight)

Let $\mathcal{A}$ be a structure and $X \subseteq \mathbb{N}$. The following are equivalent:

(1) *X is Σ^0_n relative to every copy $\mathcal{B} \cong \mathcal{A}$;*

(2) *there is $\bar{a} \in \mathcal{A}$ and a computable list of computable Σ_n formulas φ_i such that*

$$i \in X \iff \mathcal{A} \models \varphi_i(\bar{a}).$$

Theorem (Knight)

Let $\mathcal{A}$ be a structure and $X \subseteq \mathbb{N}$. The following are equivalent:

(1) *X is Σ^0_n relative to every copy $\mathcal{B} \cong \mathcal{A}$;*

(2) *there is $\bar{a} \in \mathcal{A}$ and a computable list of computable Σ_n formulas φ_i such that*

$$i \in X \iff \mathcal{A} \models \varphi_i(\bar{a}).$$

(2)$\Rightarrow$(1) is easy: Given $\mathcal{B} \cong \mathcal{A}$, we can fix $\bar{b} \in \mathcal{B}$ the image of $\bar{a}$. Hard-coding $\bar{b}$, evaluating a Σ_n formula is Σ^0_n.

For (1)$\Rightarrow$(2) we'll use computability-theoretic forcing.

For (1)$\Rightarrow$(2) we'll use computability-theoretic forcing.

If you are a:

Set theorist: Think of Cohen forcing, except meeting only countably many dense sets.

For (1)$\Rightarrow$(2) we'll use computability-theoretic forcing.

If you are a:

Set theorist: Think of Cohen forcing, except meeting only countably many dense sets.

Descriptive set theorist: Think of Vaught transforms.

For (1)$\Rightarrow$(2) we'll use computability-theoretic forcing.

If you are a:

Set theorist: Think of Cohen forcing, except meeting only countably many dense sets.

Descriptive set theorist: Think of Vaught transforms.

Model theorist: Omitting types?

The forcing conditions

Fix a structure $\mathcal{A}$.

The forcing conditions

Fix a structure $\mathcal{A}$.

We build a copy $\mathcal{G} \cong \mathcal{A}$ by building a bijection $g : \mathbb{N} \to \mathcal{A}$ and pulling back $\mathcal{A}$ along g.

The forcing conditions

Fix a structure $\mathcal{A}$.

We build a copy $\mathcal{G} \cong \mathcal{A}$ by building a bijection $g : \mathbb{N} \to \mathcal{A}$ and pulling back $\mathcal{A}$ along g.

Think of g as being built from finite injections

$$p : (0, \ldots, n-1) \to \mathbb{N}.$$

The forcing conditions

Fix a structure $\mathcal{A}$.

We build a copy $\mathcal{G} \cong \mathcal{A}$ by building a bijection $g : \mathbb{N} \to \mathcal{A}$ and pulling back $\mathcal{A}$ along g.

Think of g as being built from finite injections

$$p : (0, \ldots, n-1) \to \mathbb{N}.$$

Such a p is really just a tuple $\bar{p}$ from $\mathcal{A}$. We write $\mathcal{A}^*$ for the set of finite tuples from $\mathcal{A}$ without repetition. These are the forcing conditions.

The forcing language

The formulas of our forcing language will be written $\varphi(\dot{\mathcal{G}})$ where $\dot{\mathcal{G}}$ is a variable standing for a generic structure.

The forcing language

The formulas of our forcing language will be written $\varphi(\dot{\mathcal{G}})$ where $\dot{\mathcal{G}}$ is a variable standing for a generic structure.

The finitary formulas of our forcing language are built as follows:

The forcing language

The formulas of our forcing language will be written $\varphi(\dot{\mathcal{G}})$ where $\dot{\mathcal{G}}$ is a variable standing for a generic structure.

The finitary formulas of our forcing language are built as follows:

- $\top$ and $\bot$,

The forcing language

The formulas of our forcing language will be written $\varphi(\dot{\mathcal{G}})$ where $\dot{\mathcal{G}}$ is a variable standing for a generic structure.

The finitary formulas of our forcing language are built as follows:

- $\top$ and $\bot$,
- $R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ and $\neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ where R is a relation symbol and $a_1, \ldots, a_n \in \mathbb{N} = \mathrm{dom}(\dot{\mathcal{G}})$.

The forcing language

The formulas of our forcing language will be written $\varphi(\dot{\mathcal{G}})$ where $\dot{\mathcal{G}}$ is a variable standing for a generic structure.

The finitary formulas of our forcing language are built as follows:

- $\top$ and $\bot$,
- $R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ and $\neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ where R is a relation symbol and $a_1, \ldots, a_n \in \mathbb{N} = \mathrm{dom}(\dot{\mathcal{G}})$.
- finitary conjunctions and disjunctions of the above.

The forcing language

The formulas of our forcing language will be written $\varphi(\dot{\mathcal{G}})$ where $\dot{\mathcal{G}}$ is a variable standing for a generic structure.

The finitary formulas of our forcing language are built as follows:

- $\top$ and $\bot$,
- $R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ and $\neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ where R is a relation symbol and $a_1, \ldots, a_n \in \mathbb{N} = \mathrm{dom}(\dot{\mathcal{G}})$.
- finitary conjunctions and disjunctions of the above.

We then allow $\bigwedge\!\!\!\!\bigwedge$ and $\bigvee\!\!\!\!\bigvee$, counting quantifier complexity to stratify into classes Σ_n and Π_n.

The forcing language

The formulas of our forcing language will be written $\varphi(\dot{\mathcal{G}})$ where $\dot{\mathcal{G}}$ is a variable standing for a generic structure.

The finitary formulas of our forcing language are built as follows:

- $\top$ and $\bot$,
- $R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ and $\neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$ where R is a relation symbol and $a_1, \ldots, a_n \in \mathbb{N} = \mathrm{dom}(\dot{\mathcal{G}})$.
- finitary conjunctions and disjunctions of the above.

We then allow $\bigwedge$ and $\bigvee$, counting quantifier complexity to stratify into classes Σ_n and Π_n.

We can also define what it means for a formula to be computable.

The forcing language

We can write down formulas of the forcing language to represent computations using the generic object as an oracle.

The forcing language

We can write down formulas of the forcing language to represent computations using the generic object as an oracle.

For example, consider the statement "$\Phi^{\dot{\mathcal{G}}}(m) = n$".

The forcing language

We can write down formulas of the forcing language to represent computations using the generic object as an oracle.

For example, consider the statement "$\Phi^{\dot{\mathcal{G}}}(m) = n$".

- There are certain finite oracles which make this true.

The forcing language

We can write down formulas of the forcing language to represent computations using the generic object as an oracle.

For example, consider the statement "$\Phi^{\dot{\mathcal{G}}}(m) = n$".

- There are certain finite oracles which make this true.
- Each of these finite oracles corresponds to finitely many atomic and negated atomic facts about elements of $\dot{\mathcal{G}}$, and hence to a finitary formula of the forcing language.

The forcing language

We can write down formulas of the forcing language to represent computations using the generic object as an oracle.

For example, consider the statement "$\Phi^{\dot{\mathcal{G}}}(m) = n$".

- There are certain finite oracles which make this true.
- Each of these finite oracles corresponds to finitely many atomic and negated atomic facts about elements of $\dot{\mathcal{G}}$, and hence to a finitary formula of the forcing language.
- We can then take the disjunction of all of these formulas.

Forcing

We define $p \Vdash \varphi$ for φ a sentence of the forcing language. We begin with the finitary formulas.

Forcing

We define $p \Vdash \varphi$ for φ a sentence of the forcing language. We begin with the finitary formulas.

- $p \Vdash \top$ but $p \nVdash \bot$.

Forcing

We define $p \Vdash \varphi$ for φ a sentence of the forcing language. We begin with the finitary formulas.

- $p \Vdash \top$ but $p \nVdash \bot$.
- If $\varphi \equiv R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.

Forcing

We define $p \Vdash \varphi$ for φ a sentence of the forcing language. We begin with the finitary formulas.

- $p \Vdash \top$ but $p \nVdash \bot$.
- If $\varphi \equiv R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models \neg R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.

Forcing

We define $p \Vdash \varphi$ for φ a sentence of the forcing language. We begin with the finitary formulas.

- $p \Vdash \top$ but $p \nVdash \bot$.
- If $\varphi \equiv R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models \neg R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \psi_1 \vee \cdots \vee \psi_n$, then $p \Vdash \varphi$ if $p \Vdash \psi_i$ for some i.

Forcing

We define $p \Vdash \varphi$ for φ a sentence of the forcing language. We begin with the finitary formulas.

- $p \Vdash \top$ but $p \nVdash \bot$.
- If $\varphi \equiv R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models \neg R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \psi_1 \vee \cdots \vee \psi_n$, then $p \Vdash \varphi$ if $p \Vdash \psi_i$ for some i.
- If $\varphi \equiv \psi_1 \wedge \cdots \wedge \psi_n$, then $p \Vdash \varphi$ if $p \Vdash \psi_i$ for each i.

Forcing

For infinitary formulas:

- If $\varphi \equiv \bigvee_n \psi_n$, then $p \Vdash \varphi$ if and only if there is n such that $p \Vdash \psi_n$.

Forcing

For infinitary formulas:

- If $\varphi \equiv \bigvee_n \psi_n$, then $p \Vdash \varphi$ if and only if there is n such that $p \Vdash \psi_n$.
- If $\varphi \equiv \bigwedge_n \psi_n$, then $p \Vdash \varphi$ if and only if, for all n and $q \supseteq p$, there is $r \supseteq q$ such that $r \Vdash \psi_n$.

Forcing

For infinitary formulas:

- If $\varphi \equiv \bigvee_n \psi_n$, then $p \Vdash \varphi$ if and only if there is n such that $p \Vdash \psi_n$.
- If $\varphi \equiv \bigwedge_n \psi_n$, then $p \Vdash \varphi$ if and only if, for all n and $q \supseteq p$, there is $r \supseteq q$ such that $r \Vdash \psi_n$.

Note that this is a strong forcing relation.

Forcing

For infinitary formulas:

- If $\varphi \equiv \bigvee_n \psi_n$, then $p \Vdash \varphi$ if and only if there is n such that $p \Vdash \psi_n$.
- If $\varphi \equiv \bigwedge_n \psi_n$, then $p \Vdash \varphi$ if and only if, for all n and $q \supseteq p$, there is $r \supseteq q$ such that $r \Vdash \psi_n$.

Note that this is a strong forcing relation.

Note that the negation of a formula is not directly a formula, but is equivalent to a formula. We call this the formal negation.

Forcing lemmas

Lemma

If $p \Vdash \varphi$ *and* $q \supseteq p$, *then* $q \Vdash \varphi$.

Forcing lemmas

Lemma

If $p \Vdash \varphi$ and $q \supseteq p$, then $q \Vdash \varphi$.

Lemma

For all p, there is $q \supseteq p$ such that $q \Vdash \varphi$ or $q \Vdash \neg\varphi$.

Forcing lemmas

Lemma
If $p \Vdash \varphi$ and $q \supseteq p$, then $q \Vdash \varphi$.

Lemma
For all p, there is $q \supseteq p$ such that $q \Vdash \varphi$ or $q \Vdash \neg\varphi$.

We say that p decides φ if $p \Vdash \varphi$ or $p \Vdash \neg\varphi$.

Forcing lemmas

Lemma
If $p \Vdash \varphi$ and $q \supseteq p$, then $q \Vdash \varphi$.

Lemma
For all p, there is $q \supseteq p$ such that $q \Vdash \varphi$ or $q \Vdash \neg\varphi$.

We say that p decides φ if $p \Vdash \varphi$ or $p \Vdash \neg\varphi$.

Lemma
It is not the case that $p \Vdash \varphi$ and $p \Vdash \neg\varphi$.

Generics

Given a countable collection of formulas of the forcing language,

$$g : \mathbb{N} \to \mathcal{A}$$

is generic (for that collection) if it decides each formula, i.e., for each such φ there is $p \subseteq g$ such that p decides φ.

Generics

Given a countable collection of formulas of the forcing language,

$$g : \mathbb{N} \to \mathcal{A}$$

is generic (for that collection) if it decides each formula, i.e., for each such φ there is $p \subseteq g$ such that p decides φ.

For any condition p, a generic $g \supseteq p$ exists.

Generics

Given a countable collection of formulas of the forcing language,

$$g : \mathbb{N} \to \mathcal{A}$$

is generic (for that collection) if it decides each formula, i.e., for each such φ there is $p \subseteq g$ such that p decides φ.

For any condition p, a generic $g \supseteq p$ exists.

Given such a generic, let $\mathcal{G} = g^{-1}(\mathcal{A})$.

Generics

Given a countable collection of formulas of the forcing language,

$$g : \mathbb{N} \to \mathcal{A}$$

is generic (for that collection) if it decides each formula, i.e., for each such φ there is $p \subseteq g$ such that p decides φ.

For any condition p, a generic $g \supseteq p$ exists.

Given such a generic, let $\mathcal{G} = g^{-1}(\mathcal{A})$.

Lemma

If g is generic and $\mathcal{G} = g^{-1}(\mathcal{A})$, then $\varphi(\mathcal{G})$ is true if and only if there is $p \subseteq g$ such that $p \Vdash \varphi(\dot{\mathcal{G}})$.

Definability of forcing

Lemma

Given a computable Σ_n formula φ in the forcing language there are computable formulas $\mathrm{Force}_{\varphi,n}(x_1, \ldots, x_n)$ *such that*

$$p \Vdash \varphi \iff \mathcal{A} \models \mathrm{Force}_{\varphi,n}(p).$$

Definability of forcing

Lemma

Given a computable Σ_n formula φ in the forcing language there are computable formulas $\mathrm{Force}_{\varphi,n}(x_1, \ldots, x_n)$ *such that*

$$p \Vdash \varphi \iff \mathcal{A} \models \mathrm{Force}_{\varphi,n}(p).$$

Argue inductively, considering the cases:

- If $\varphi \equiv R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models \neg R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.

Definability of forcing

Lemma

Given a computable Σ_n formula φ in the forcing language there are computable formulas $\mathrm{Force}_{\varphi,n}(x_1, \ldots, x_n)$ *such that*

$$p \Vdash \varphi \iff \mathcal{A} \models \mathrm{Force}_{\varphi,n}(p).$$

Argue inductively, considering the cases:

- If $\varphi \equiv R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models \neg R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \bigvee_n \psi_n$, then $p \Vdash \varphi$ if and only if there is n such that $p \Vdash \psi_n$.

Definability of forcing

Lemma

Given a computable Σ_n formula φ in the forcing language there are computable formulas $\mathrm{Force}_{\varphi,n}(x_1, \ldots, x_n)$ *such that*

$$p \Vdash \varphi \iff \mathcal{A} \models \mathrm{Force}_{\varphi,n}(p).$$

Argue inductively, considering the cases:

- If $\varphi \equiv R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \neg R^{\dot{\mathcal{G}}}(a_1, \ldots, a_n)$, then $p \Vdash \varphi$ if $\bar{p}(a_1), \ldots, \bar{p}(a_n)$ are defined and $\mathcal{A} \models \neg R(\bar{p}(a_1), \ldots, \bar{p}(a_n))$.
- If $\varphi \equiv \bigvee_n \psi_n$, then $p \Vdash \varphi$ if and only if there is n such that $p \Vdash \psi_n$.
- If $\varphi \equiv \bigwedge_n \psi_n$, then $p \Vdash \varphi$ if and only if, for all n and $q \supseteq p$, there is $r \supseteq q$ such that $r \Vdash \psi_n$.

Theorem (Knight)

Let $\mathcal{A}$ be a structure and $X \subseteq \mathbb{N}$. The following are equivalent:

(1) *X is Σ^0_n relative to every copy $\mathcal{B} \cong \mathcal{A}$;*

(2) *there is $\bar{a} \in \mathcal{A}$ and a computable list of computable Σ_n formulas φ_i such that*

$$i \in X \iff \mathcal{A} \models \varphi_i(\bar{a}).$$

Theorem (Knight)

Let $\mathcal{A}$ be a structure and $X \subseteq \mathbb{N}$. The following are equivalent:

(1) *X is Σ^0_n relative to every copy $\mathcal{B} \cong \mathcal{A}$;*

(2) *there is $\bar{a} \in \mathcal{A}$ and a computable list of computable Σ_n formulas φ_i such that*

$$i \in X \iff \mathcal{A} \models \varphi_i(\bar{a}).$$

Let's prove (1)⇒(2).

Let $g : \mathcal{G} \to \mathcal{A}$ be a sufficiently generic copy of $\mathcal{A}$.

Let $g : \mathcal{G} \to \mathcal{A}$ be a sufficiently generic copy of $\mathcal{A}$.

There are formulas $\varphi_e(\dot{\mathcal{G}})$ that expresses that X is Σ^0_n relative to $\dot{\mathcal{G}}$ using the eth Σ^0_n operator Γ_e.

Let $g : \mathcal{G} \to \mathcal{A}$ be a sufficiently generic copy of $\mathcal{A}$.

There are formulas $\varphi_e(\dot{\mathcal{G}})$ that expresses that X is Σ^0_n relative to $\dot{\mathcal{G}}$ using the eth Σ^0_n operator Γ_e.

Since X is Σ^0_n relative to $\mathcal{G}$, say $X = \Gamma^{\mathcal{G}}_e$, $\varphi_e(\mathcal{G})$ is true.

Let $g : \mathcal{G} \to \mathcal{A}$ be a sufficiently generic copy of $\mathcal{A}$.

There are formulas $\varphi_e(\dot{\mathcal{G}})$ that expresses that X is Σ^0_n relative to $\dot{\mathcal{G}}$ using the eth Σ^0_n operator Γ_e.

Since X is Σ^0_n relative to $\mathcal{G}$, say $X = \Gamma^{\mathcal{G}}_e$, $\varphi_e(\mathcal{G})$ is true.

So there is $p \subseteq g$ with $p \Vdash \varphi(\dot{\mathcal{G}})$.

Let $g : \mathcal{G} \to \mathcal{A}$ be a sufficiently generic copy of $\mathcal{A}$.

There are formulas $\varphi_e(\dot{\mathcal{G}})$ that expresses that X is Σ^0_n relative to $\dot{\mathcal{G}}$ using the eth Σ^0_n operator Γ_e.

Since X is Σ^0_n relative to $\mathcal{G}$, say $X = \Gamma^{\mathcal{G}}_e$, $\varphi_e(\mathcal{G})$ is true.

So there is $p \subseteq g$ with $p \Vdash \varphi(\dot{\mathcal{G}})$.

Consider the computable list of Σ^0_n formulas which say that $n \in \Gamma^{\dot{\mathcal{G}}}_e$.

Let $g : \mathcal{G} \to \mathcal{A}$ be a sufficiently generic copy of $\mathcal{A}$.

There are formulas $\varphi_e(\dot{\mathcal{G}})$ that expresses that X is Σ^0_n relative to $\dot{\mathcal{G}}$ using the eth Σ^0_n operator Γ_e.

Since X is Σ^0_n relative to $\mathcal{G}$, say $X = \Gamma^{\mathcal{G}}_e$, $\varphi_e(\mathcal{G})$ is true.

So there is $p \subseteq g$ with $p \Vdash \varphi(\dot{\mathcal{G}})$.

Consider the computable list of Σ^0_n formulas which say that $n \in \Gamma^{\dot{\mathcal{G}}}_e$.

We claim that $n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma^{\dot{\mathcal{G}}}_e$.

We claim that $n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.

We claim that $n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.

- If $n \in X$, then $\Gamma_e^{\mathcal{G}}$, and there is $g \supset q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.

We claim that $n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.

- If $n \in X$, then $\Gamma_e^{\mathcal{G}}$, and there is $g \supset q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.
- If there is $q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$, then there is a generic $g' \supset q \supseteq p$ and $g' : \mathcal{G}' \to \mathcal{A}$.

We claim that $n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.

- If $n \in X$, then $\Gamma_e^{\mathcal{G}}$, and there is $g \supset q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.
- If there is $q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$, then there is a generic $g' \supset q \supseteq p$ and $g' : \mathcal{G}' \to \mathcal{A}$.

  Since $g' \supseteq p$, $\Gamma_e^{\mathcal{G}'} = X$.

We claim that $n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.

- If $n \in X$, then $\Gamma_e^{\mathcal{G}}$, and there is $g \supset q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.
- If there is $q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$, then there is a generic $g' \supset q \supseteq p$ and $g' : \mathcal{G}' \to \mathcal{A}$.

  Since $g' \supseteq p$, $\Gamma_e^{\mathcal{G}'} = X$.

  Since $g' \supseteq q$, $n \in \Gamma_e^{\mathcal{G}'}$.

We claim that $n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.

- If $n \in X$, then $\Gamma_e^{\mathcal{G}}$, and there is $g \supset q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$.
- If there is $q \supseteq p$ such that $q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}$, then there is a generic $g' \supset q \supseteq p$ and $g' : \mathcal{G}' \to \mathcal{A}$.

  Since $g' \supseteq p$, $\Gamma_e^{\mathcal{G}'} = X$.

  Since $g' \supseteq q$, $n \in \Gamma_e^{\mathcal{G}'}$.

  Thus $n \in X$.

We have

$$n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_{\mathbf{e}}^{\dot{\mathcal{G}}}.$$

We have

$$n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}.$$

Since $n \in \Gamma_e^{\dot{\mathcal{G}}}$ is a computable Σ_n, by definability of forcing,

$$n \in X \iff \exists q \supseteq p \quad \mathrm{Force}_{n \in \Gamma_e^{\dot{\mathcal{G}}}}(q).$$

We have

$$n \in X \iff \exists q \supseteq p \quad q \Vdash n \in \Gamma_e^{\dot{\mathcal{G}}}.$$

Since $n \in \Gamma_e^{\dot{\mathcal{G}}}$ is a computable Σ_n, by definability of forcing,

$$n \in X \iff \exists q \supseteq p \quad \text{Force}_{n \in \Gamma_e^{\dot{\mathcal{G}}}}(q).$$

The right-hand-side is a computable sequence of Σ_n formulas with paramters p.

Definition

Let $\mathcal{A}$ be a structure and $R \subseteq \mathcal{A}^n$.

R is relatively intrinsically Σ^0_n if whenever $g : \mathcal{A} \cong \mathcal{B}$, $g(R) \subseteq \mathcal{B}^n$ is Σ^0_n relative to $\mathcal{B}$.

Definition

Let $\mathcal{A}$ be a structure and $R \subseteq \mathcal{A}^n$.

R is relatively intrinsically Σ^0_n if whenever $g : \mathcal{A} \cong \mathcal{B}$, $g(R) \subseteq \mathcal{B}^n$ is Σ^0_n relative to $\mathcal{B}$.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

R is relatively intrinsically Σ^0_n if and only if it is definable in $\mathcal{A}$ by a computable Σ_n formula over a finite tuple of parameters.

Definition

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if whenever $\mathcal{B} \cong \mathcal{A}$ there is an isomorphism $g : \mathcal{B} \to \mathcal{A}$ which is Δ^0_n relative to $\mathcal{A}$ and $\mathcal{B}$.

Definition

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if whenever $\mathcal{B} \cong \mathcal{A}$ there is an isomorphism $g : \mathcal{B} \to \mathcal{A}$ which is Δ^0_n relative to $\mathcal{A}$ and $\mathcal{B}$.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if and only if $\mathcal{A}$ has a c.e. Scott family of computable Σ_n formulas over finitely many parameters.

Definition

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if whenever $\mathcal{B} \cong \mathcal{A}$ there is an isomorphism $g : \mathcal{B} \to \mathcal{A}$ which is Δ^0_n relative to $\mathcal{A}$ and $\mathcal{B}$.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if and only if $\mathcal{A}$ has a c.e. Scott family of computable Σ_n formulas over finitely many parameters.

A Scott family is a collection of formulas S such that

Definition

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if whenever $\mathcal{B} \cong \mathcal{A}$ there is an isomorphism $g : \mathcal{B} \to \mathcal{A}$ which is Δ^0_n relative to $\mathcal{A}$ and $\mathcal{B}$.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if and only if $\mathcal{A}$ has a c.e. Scott family of computable Σ_n formulas over finitely many parameters.

A Scott family is a collection of formulas S such that

- each tuple from $\mathcal{A}$ satisfies at least one formula from S, and

Definition

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if whenever $\mathcal{B} \cong \mathcal{A}$ there is an isomorphism $g : \mathcal{B} \to \mathcal{A}$ which is Δ^0_n relative to $\mathcal{A}$ and $\mathcal{B}$.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

$\mathcal{A}$ is relatively intrinsically Δ^0_n-categorical if and only if $\mathcal{A}$ has a c.e. Scott family of computable Σ_n formulas over finitely many parameters.

A Scott family is a collection of formulas S such that

- each tuple from $\mathcal{A}$ satisfies at least one formula from S, and
- if two tuples from $\mathcal{A}$ satisfy the same formula from S, then they are automorphic.

The parameters come from the non-uniformity. There are also uniform versions without parameters.

The parameters come from the non-uniformity. There are also uniform versions without parameters.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

R is uniformly relatively intrinsically Σ^0_n if and only if it is definable in $\mathcal{A}$ by a computable Σ_n formula without parameters.

The parameters come from the non-uniformity. There are also uniform versions without parameters.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

R is uniformly relatively intrinsically Σ^0_n if and only if it is definable in $\mathcal{A}$ by a computable Σ_n formula without parameters.

Theorem (Ash-Knight-Manasse-Slaman, Chisholm)

$\mathcal{A}$ is uniformly relatively intrinsically Δ^0_n-categorical if and only if $\mathcal{A}$ has a c.e. Scott family of Σ_n formulas without parameters.

Definition

$\mathrm{Mod}(\mathcal{L})$ is the Polish space of all presentations of $\mathcal{L}$-structures with domain $\mathbb{N}$.

Definition

$\mathrm{Mod}(\mathcal{L})$ is the Polish space of all presentations of $\mathcal{L}$-structures with domain $\mathbb{N}$.

The basic clopen sets are given by atomic or negated atomic facts,

$$[\varphi(n_1, \ldots, n_k)] = \{\mathcal{A} \in \mathrm{Mod}(\mathcal{L}) : \mathcal{A} \models \varphi(n_1, \ldots, n_k)\}.$$

Definition

$\mathrm{Mod}(\mathcal{L})$ is the Polish space of all presentations of $\mathcal{L}$-structures with domain $\mathbb{N}$.

The basic clopen sets are given by atomic or negated atomic facts,

$$[\varphi(n_1, \ldots, n_k)] = \{\mathcal{A} \in \mathrm{Mod}(\mathcal{L}) : \mathcal{A} \models \varphi(n_1, \ldots, n_k)\}.$$

A set $A \subseteq \mathrm{Mod}(\mathcal{L})$ is *invariant* if it is closed under isomorphism.

Theorem (Lopez-Escobar, Vaught, vanden Boom)

Let A be an invariant set of $\mathrm{Mod}(\mathcal{L})$. *Then A is:*

Theorem (Lopez-Escobar, Vaught, vanden Boom)

Let A be an invariant set of $\mathrm{Mod}(\mathcal{L})$. *Then A is:*

- *Borel if and only if it is definable by a formula;*

Theorem (Lopez-Escobar, Vaught, vanden Boom)

Let A be an invariant set of $\mathrm{Mod}(\mathcal{L})$*. Then A is:*

- *Borel if and only if it is definable by a formula;*
- $\mathbf{\Pi}^0_\alpha$ *if and only if it is definable by a* Π_α *formula;*

Theorem (Lopez-Escobar, Vaught, vanden Boom)

Let A be an invariant set of $\mathrm{Mod}(\mathcal{L})$. *Then A is:*

- *Borel if and only if it is definable by a formula;*
- $\mathbf{\Pi}^0_\alpha$ *if and only if it is definable by a* Π_α *formula;*
- Π^0_α *if and only if it is definable by a computable* Π_α *formula.*

Definition

A theory T is relatively decidable if for all $\mathcal{A} \models T$, T together with the atomic diagram of $\mathcal{A}$ compute the elementary diagram of $\mathcal{A}$.

Definition

A theory T is relatively decidable if for all $\mathcal{A} \models T$, T together with the atomic diagram of $\mathcal{A}$ compute the elementary diagram of $\mathcal{A}$.

Example

If T is model complete, then every model of T is relatively decidable: every formula is equivalent, modulo T, to both an existential formula and a universal formula.

Definition

A theory T is relatively decidable if for all $\mathcal{A} \models T$, T together with the atomic diagram of $\mathcal{A}$ compute the elementary diagram of $\mathcal{A}$.

Example

If T is model complete, then every model of T is relatively decidable: every formula is equivalent, modulo T, to both an existential formula and a universal formula.

In fact, model completeness characterizes the uniform version:

Definition

A theory T is relatively decidable if for all $\mathcal{A} \models T$, T together with the atomic diagram of $\mathcal{A}$ compute the elementary diagram of $\mathcal{A}$.

Example

If T is model complete, then every model of T is relatively decidable: every formula is equivalent, modulo T, to both an existential formula and a universal formula.

In fact, model completeness characterizes the uniform version:

Theorem (Chubb, Miller, Solomon)

A theory T is uniformly relatively decidable if and only if it is model complete.

The non-uniform case does not follow so easily. But it is true.

The non-uniform case does not follow so easily. But it is true.

Theorem (Harrison-Trainor, Tan)

A complete theory T is relatively decidable if and only if there is a formula $\varphi(\bar{x})$ such that

The non-uniform case does not follow so easily. But it is true.

Theorem (Harrison-Trainor, Tan)

A complete theory T is relatively decidable if and only if there is a formula $\varphi(\bar{x})$ such that

1. $T \models \exists \bar{x} \varphi(\bar{x})$, *and*

The non-uniform case does not follow so easily. But it is true.

Theorem (Harrison-Trainor, Tan)

A complete theory T is relatively decidable if and only if there is a formula $\varphi(\bar{x})$ such that

1. $T \models \exists \bar{x} \varphi(\bar{x})$, *and*
2. $T \cup \{\varphi(\bar{c})\}$ *is model complete.*

The non-uniform case does not follow so easily. But it is true.

Theorem (Harrison-Trainor, Tan)

A complete theory T is relatively decidable if and only if there is a formula $\varphi(\bar{x})$ such that

1. $T \models \exists \bar{x} \varphi(\bar{x})$, *and*
2. $T \cup \{\varphi(\bar{c})\}$ *is model complete.*

This characterization does not extend to incomplete theories.

Theorem (HT, Miller, Montalbán; Chen)

Let T be an $\mathcal{L}$-sentence and let T' be an $\mathcal{L}'$-sentence.

Every Borel functor from models of T' to models of T is Borel naturally isomorphic to the functor induced by an interpretation of $(\mathcal{L}, T)$ in $(\mathcal{L}', T')$.

Theorem (HT, Miller, Montalbán; Chen)

Let T be an $\mathcal{L}$-sentence and let T' be an $\mathcal{L}'$-sentence.

Every Borel functor from models of T' to models of T is Borel naturally isomorphic to the functor induced by an interpretation of $(\mathcal{L}, T)$ in $(\mathcal{L}', T')$.

Example

The map from a ring R to the polynomial ring $R[x]$ is functorial and Borel. It is induced by an infinitary interpretation of $R[x]$ in R.

Theorem (HT, Miller, Montalbán; Chen)

Let T be an $\mathcal{L}$-sentence and let T' be an $\mathcal{L}'$-sentence.

Every Borel functor from models of T' to models of T is Borel naturally isomorphic to the functor induced by an interpretation of $(\mathcal{L}, T)$ in $(\mathcal{L}', T')$.

Example

The map from a ring R to the polynomial ring $R[x]$ is functorial and Borel. It is induced by an infinitary interpretation of $R[x]$ in R.

Infinitary interpretations can use tuples of all sizes as part of the domain.

To get a correspondence between computability and definability, you need to consider non-computable copies.

To get a correspondence between computability and definability, you need to consider non-computable copies.

Theorem (Downey, Kach, Lempp, Lewis-Pye, Montalbán, Turetsky)

There is no characterization of the computable structures which are computably categorical.

To get a correspondence between computability and definability, you need to consider non-computable copies.

Theorem (Downey, Kach, Lempp, Lewis-Pye, Montalbán, Turetsky)

There is no characterization of the computable structures which are computably categorical.

(The set of indices of such structures is Π^1_1-complete.)

To get a correspondence between computability and definability, you need to consider non-computable copies.

Theorem (Downey, Kach, Lempp, Lewis-Pye, Montalbán, Turetsky)

There is no characterization of the computable structures which are computably categorical.

(The set of indices of such structures is Π^1_1-complete.)

So relative computable categoricity has a characterization, but (plain) computable categoricity does not.

You cannot always get a correspondence between computability and definability. Here are two non-theorems (with provable counterexamples).

You cannot always get a correspondence between computability and definability. Here are two non-theorems (with provable counterexamples).

False Statement (Kalociński, Slaman)

The following are equivalent:

- *Every automorphism of $\mathcal{A}$ is computable.*

You cannot always get a correspondence between computability and definability. Here are two non-theorems (with provable counterexamples).

False Statement (Kalociński, Slaman)

The following are equivalent:

- *Every automorphism of $\mathcal{A}$ is computable.*
- *There is a finite set of parameters $\bar{a}$ such that every automorphism f of $\mathcal{A}$ is Σ_1-definable over $\bar{a}, f(\bar{a})$.*

An automorphism f of $\mathcal{A}$ is Σ_1-definable over $\bar{b}$ if there is a computable Σ_1 formulas $\varphi(x, y, \bar{b})$ such that $f(x) = y$ if and only if $\mathcal{A} \models \varphi(x, y, \bar{b})$.

False Statement (HT, Montalbán)

Let $\mathcal{F} = \{X_0, X_1, \ldots\}$ be a family of subsets of $\mathbb{N}$. Let $\mathcal{A}$ be a structure. The following are equivalent:

False Statement (HT, Montalbán)

Let $\mathcal{F} = \{X_0, X_1, \ldots\}$ be a family of subsets of $\mathbb{N}$. Let $\mathcal{A}$ be a structure. The following are equivalent:

- *Every copy of $\mathcal{A}$ enumerates a listing of $\mathcal{F}$,*

False Statement (HT, Montalbán)

Let $\mathcal{F} = \{X_0, X_1, \ldots\}$ be a family of subsets of $\mathbb{N}$. Let $\mathcal{A}$ be a structure. The following are equivalent:

- *Every copy of $\mathcal{A}$ enumerates a listing of $\mathcal{F}$,*
- *There are finite parameters $\bar{a}$, a computable list of Σ_1 formulas $\varphi_n(\bar{x}, \bar{a})$, and a computable list of Σ_1 formulas $\psi_{n,m}(\bar{x}, \bar{a})$ such that*

$$\{\{m : \mathcal{A} \models \psi_{n,m}(\bar{x}, \bar{a})\} : \mathcal{A} \models \varphi_n(\bar{x}, \bar{a})\}$$

is a listing of $\mathcal{F}$.

False Statement (HT, Montalbán)

Let $\mathcal{F} = \{X_0, X_1, \ldots\}$ be a family of subsets of $\mathbb{N}$. Let $\mathcal{A}$ be a structure. The following are equivalent:

- *Every copy of $\mathcal{A}$ enumerates a listing of $\mathcal{F}$,*
- *There are finite parameters $\bar{a}$, a computable list of Σ_1 formulas $\varphi_n(\bar{x}, \bar{a})$, and a computable list of Σ_1 formulas $\psi_{n,m}(\bar{x}, \bar{a})$ such that*

$$\{\{m : \mathcal{A} \models \psi_{n,m}(\bar{x}, \bar{a})\} : \mathcal{A} \models \varphi_n(\bar{x}, \bar{a})\}$$

is a listing of $\mathcal{F}$.

(This follows from the fact that there is a structure whose tree of tuples is computable but the structure has no computable copy.)

In Part 3, we'll move further away from computability theory and look at some questions of pure infinitary logic which can still be considered part of computable structure theory.

In Part 3, we'll move further away from computability theory and look at some questions of pure infinitary logic which can still be considered part of computable structure theory.

At the end of Part 3, we'll see how this all ties back to computation.