Results in Computable Model Theory of Continuous Logic

ASL North American Annual Meeting
Online (prev. UC Irvine)
March 26, 2020

Caleb Camrud

Iowa State University

Plan

(i) Preliminaries on Continuous Logic.

Plan

(i) Preliminaries on Continuous Logic.

(ii) Effectivizing Continuous Logic.

Plan

(i) Preliminaries on Continuous Logic.

(ii) Effectivizing Continuous Logic.

(iii) Effective Completeness Theorem

Plan

(i) Preliminaries on Continuous Logic.

(ii) Effectivizing Continuous Logic.

(iii) Effective Completeness Theorem

(iv) Computational vs. Syntactic Complexity

Preliminaries on Continuous Logic

The language of first-order logic contains:

Preliminaries on Continuous Logic

The language of first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, \ldots\}$

Preliminaries on Continuous Logic

The language of first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, ...\}$

Connective symbols: $\{\neg, \wedge, \vee, \rightarrow, ...\}$

Preliminaries on Continuous Logic

The language of first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, ...\}$

Connective symbols: $\{\neg, \wedge, \vee, \rightarrow, ...\}$

Quantifier symbols: $\{\forall, \exists\}$

Preliminaries on Continuous Logic

The language of first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, ...\}$

Connective symbols: $\{\neg, \wedge, \vee, \rightarrow, ...\}$

Quantifier symbols: $\{\forall, \exists\}$

Binary relation symbol: $\{=\}$

Preliminaries on Continuous Logic

So the well-formed formulas look something like:

Preliminaries on Continuous Logic

So the well-formed formulas look something like:

$\exists x_7,\ x_7 = x_8$

Preliminaries on Continuous Logic

So the well-formed formulas look something like:

$\exists x_7,\ x_7 = x_8$

$\forall x_1\ \exists x_2\ (P(x_1) \wedge Q(x_2)) \rightarrow R(x_3)$

Preliminaries on Continuous Logic

So the well-formed formulas look something like:

$\exists x_7,\ x_7 = x_8$

$\forall x_1\ \exists x_2\ (P(x_1) \wedge Q(x_2)) \rightarrow R(x_3)$

$\neg(P(x_0) \wedge (\neg P(x_0)))$

Preliminaries on Continuous Logic

These work great in the discrete case!

Preliminaries on Continuous Logic

These work great in the discrete case!

But representing continuous structures in this setting can be tedious, and sometimes impossible.

Preliminaries on Continuous Logic

The language of continuous first-order logic contains:

Preliminaries on Continuous Logic

The language of continuous first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, \ldots\}$

Preliminaries on Continuous Logic

The language of continuous first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, ...\}$

Connective symbols: One for every function in the set $\{u : [0,1]^n \to [0,1] \mid u \text{ is continuous }\}$

Preliminaries on Continuous Logic

The language of continuous first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, ...\}$

Connective symbols: One for every function in the set $\{u : [0,1]^n \to [0,1] \mid u \text{ is continuous } \}$

Quantifier symbols: $\{\sup, \inf\}$

Preliminaries on Continuous Logic

The language of continuous first-order logic contains:

Variable symbols: $\{x_0, x_1, x_2, ...\}$

Connective symbols: One for every function in the set $\{u : [0,1]^n \to [0,1] \mid u \text{ is continuous } \}$

Quantifier symbols: $\{\sup, \inf\}$

Binary predicate symbol: $\{d(\cdot,\cdot)\}$

Preliminaries on Continuous Logic

So well-formed formulas of continuous first-order logic look like:

Preliminaries on Continuous Logic

So well-formed formulas of continuous first-order logic look like:

$\inf_{x_1,x_2}(d(x_1, x_2))$

Preliminaries on Continuous Logic

So well-formed formulas of continuous first-order logic look like:

$\inf_{x_1,x_2}(d(x_1,x_2))$

$\sup_{x_0}(1-(d(x_0,0)\vee(1-d(x_0,0))))$

Preliminaries on Continuous Logic

The **truth values** of continuous logic is the set $[0, 1]$.

Preliminaries on Continuous Logic

The **truth values** of continuous logic is the set $[0, 1]$. 0 will indicate truth, and 1 falsehood.

Preliminaries on Continuous Logic

The **truth values** of continuous logic is the set $[0, 1]$. 0 will indicate truth, and 1 falsehood.

N.B. Defining $X = \{0, 1\}$ and d as the discrete metric, continuous logic collapses to classical first-order logic.

Preliminaries on Continuous Logic

The **truth values** of continuous logic is the set $[0, 1]$. 0 will indicate truth, and 1 falsehood.

N.B. Defining $X = \{0, 1\}$ and d as the discrete metric, continuous logic collapses to classical first-order logic. In this sense, it can be thought of as a generalization.

Preliminaries on Continuous Logic

The **truth values** of continuous logic is the set $[0, 1]$. 0 will indicate truth, and 1 falsehood.

N.B. Defining $X = \{0, 1\}$ and d as the discrete metric, continuous logic collapses to classical first-order logic. In this sense, it can be thought of as a generalization.

Unlike classical logic, since 0 represents truth and 1 falsehood, $\vee$ interpreted as the maximum operator acts as an 'and' gate, and $\wedge$ interpreted as the minimum operator acts as an 'or' gate.

Preliminaries on Continuous Logic

Let (M, d) be a complete, bounded metric space.

Preliminaries on Continuous Logic

Let (M, d) be a complete, bounded metric space.

A **predicate** R on M is a uniformly continuous function $R : M^n \to [0, 1]$ which somehow *describes* the objects of the space, assigning a truth value to its inputs.

Preliminaries on Continuous Logic

Let (M, d) be a complete, bounded metric space.

A **predicate** R on M is a uniformly continuous function $R : M^n \to [0,1]$ which somehow *describes* the objects of the space, assigning a truth value to its inputs.

Ex: The metric d can be thought of as a binary predicate, since it describes how far away two objects in M are. If $d(x, y) = 0$, then it's "true" that they equal each other.

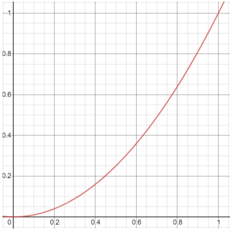
Preliminaries on Continuous Logic

A **function** F on M is a uniformly continuous function $F : M^n \to M$; i.e. it takes in n-many points of M and outputs some other unique point in M.

Preliminaries on Continuous Logic

A **function** F on M is a uniformly continuous function $F : M^n \to M$; i.e. it takes in n-many points of M and outputs some other unique point in M.

Ex: $f(x) = x^2$ is a function from $[0, 1] \to [0, 1]$.



Preliminaries on Continuous Logic

A **metric structure** $\mathfrak{M}$ on (M, d) is a tuple

$$\mathfrak{M} = (M, d, R_i, F_j, a_k : i \in I, j \in J, k \in K)$$

where the R's are predicates, the F's are functions, and the a's are designated points.

Preliminaries on Continuous Logic

A **metric structure** $\mathfrak{M}$ on (M, d) is a tuple

$$\mathfrak{M} = (M, d, R_i, F_j, a_k : i \in I, j \in J, k \in K)$$

where the R's are predicates, the F's are functions, and the a's are designated points.

Ex: $\mathfrak{M} = ([0,1], |\cdot|, P_{\mathbb{Q}} : [0,1] \to [0,1], \mathbb{Q} \cap [0,1])$ where $|\cdot|$ denotes the Euclidean metric and $P_{\mathbb{Q}} : [0,1] \to [0,1]$ rational polynomials whose range is bounded by $[0,1]$.

Preliminaries on Continuous Logic

A **signature** over a metric structure $\mathfrak{M}$ is a language $\mathcal{L}$ such that:

Preliminaries on Continuous Logic

A **signature** over a metric structure $\mathfrak{M}$ is a language $\mathcal{L}$ such that:

For every predicate R of $\mathfrak{M}$ there is a **predicate symbol** P, integer $n(P) = n(R)$, closed bounded interval I_P defined as the range of R, and modulus of uniform continuity Δ_P;

Preliminaries on Continuous Logic

A **signature** over a metric structure $\mathfrak{M}$ is a language $\mathcal{L}$ such that:

For every predicate R of $\mathfrak{M}$ there is a **predicate symbol** P, integer $n(P) = n(R)$, closed bounded interval I_P defined as the range of R, and modulus of uniform continuity Δ_P;

For every function F of $\mathfrak{M}$ there is a **function symbol** f, integer $n(f) = n(F)$, and modulus of uniform continuity Δ_f;

Preliminaries on Continuous Logic

A **signature** over a metric structure $\mathfrak{M}$ is a language $\mathcal{L}$ such that:

For every predicate R of $\mathfrak{M}$ there is a **predicate symbol** P, integer $n(P) = n(R)$, closed bounded interval I_P defined as the range of R, and modulus of uniform continuity Δ_P;

For every function F of $\mathfrak{M}$ there is a **function symbol** f, integer $n(f) = n(F)$, and modulus of uniform continuity Δ_f;

For each distinguished point a there is a **constant symbol** t;

Preliminaries on Continuous Logic

A **signature** over a metric structure $\mathfrak{M}$ is a language $\mathcal{L}$ such that:

For every predicate R of $\mathfrak{M}$ there is a **predicate symbol** P, integer $n(P) = n(R)$, closed bounded interval I_P defined as the range of R, and modulus of uniform continuity Δ_P;

For every function F of $\mathfrak{M}$ there is a **function symbol** f, integer $n(f) = n(F)$, and modulus of uniform continuity Δ_f;

For each distinguished point a there is a **constant symbol** t;

And a non-negative real number $D_{\mathcal{L}} = \sup\{d(a, b) : a, b \in M\}$, called the **diameter**.

Preliminaries on Continuous Logic

The set of **terms** of $\mathcal{L}$ is "built up" in the standard manner from variable symbols, constant symbols, and functions (symbols) attached to previous terms.

Preliminaries on Continuous Logic

The set of **terms** of $\mathcal{L}$ is "built up" in the standard manner from variable symbols, constant symbols, and functions (symbols) attached to previous terms.

Ex: Let $\mathfrak{M}$ be the structure given previously, and suppose $\mathcal{L}$ is a signature over $\mathfrak{M}$.

Preliminaries on Continuous Logic

The set of **terms** of $\mathcal{L}$ is "built up" in the standard manner from variable symbols, constant symbols, and functions (symbols) attached to previous terms.

Ex: Let $\mathfrak{M}$ be the structure given previously, and suppose $\mathcal{L}$ is a signature over $\mathfrak{M}$.

$`x_1', `\frac{1}{3}', `\left((x_4)^2 + \frac{1}{2}\right) \upharpoonright_{[0,1]}\ ' \in \mathrm{Term}_{\mathcal{L}}.$

Preliminaries on Continuous Logic

The set of **atomic formulas** of $\mathcal{L}$, denoted $\mathrm{Atom}_{\mathcal{L}}$, contains expressions of the form $P(t_1, ..., t_n)$ where P is a predicate symbol of $\mathcal{L}$ and $t_1, ..., t_n$ are terms. (Recall that d is a binary predicate symbol of $\mathcal{L}$.)

Preliminaries on Continuous Logic

The set of **atomic formulas** of $\mathcal{L}$, denoted $\mathrm{Atom}_{\mathcal{L}}$, contains expressions of the form $P(t_1, ..., t_n)$ where P is a predicate symbol of $\mathcal{L}$ and $t_1, ..., t_n$ are terms. (Recall that d is a binary predicate symbol of $\mathcal{L}$.)

Ex: '$|x_1 - 0|$', '$\left|\frac{2}{5} - \frac{1}{3}\right|$' $\in \mathrm{Atom}_{\mathcal{L}}$.

Preliminaries on Continuous Logic

The set of **well-formed formulas** of $\mathcal{L}$, denoted $W_{\mathcal{L}}$, is defined as the smallest set of expressions s.t.

Preliminaries on Continuous Logic

The set of **well-formed formulas** of $\mathcal{L}$, denoted $W_{\mathcal{L}}$, is defined as the smallest set of expressions s.t.

(i) any atomic formula of $\mathcal{L}$ is a well-formed formula of $\mathcal{L}$;

Preliminaries on Continuous Logic

The set of **well-formed formulas** of $\mathcal{L}$, denoted $W_{\mathcal{L}}$, is defined as the smallest set of expressions s.t.

(i) any atomic formula of $\mathcal{L}$ is a well-formed formula of $\mathcal{L}$;

(ii) if $u : [0,1]^n \to [0,1]$ is continuous and $\varphi_1, ..., \varphi_n$ are well-formed formulas, then $\mathbf{u}(\varphi_1, ..., \varphi_n)$ is a well-formed formula;

Preliminaries on Continuous Logic

The set of **well-formed formulas** of $\mathcal{L}$, denoted $W_{\mathcal{L}}$, is defined as the smallest set of expressions s.t.

(i) any atomic formula of $\mathcal{L}$ is a well-formed formula of $\mathcal{L}$;

(ii) if $u : [0,1]^n \to [0,1]$ is continuous and $\varphi_1, \ldots, \varphi_n$ are well-formed formulas, then $\mathbf{u}(\varphi_1, \ldots, \varphi_n)$ is a well-formed formula;

(iii) if φ is a well-formed formula and x is a variable symbol, then '$\sup$'$_x\varphi$ and '$\inf$'$_x\varphi$ are well-formed formulas (normally we'll leave off the quotation marks).

Preliminaries on Continuous Logic

$\mathrm{lh}(\varphi)$ is the **length** of φ; i.e. count how many (not necessarily distinct) symbols appear in φ.

Preliminaries on Continuous Logic

$\mathrm{lh}(\varphi)$ is the **length** of φ; i.e. count how many (not necessarily distinct) symbols appear in φ.

A wff $\varphi \in W_{\mathcal{L}}$ is **quantifier-free** if no quantifiers ($\sup_x$ or $\inf_x$) appear in φ.

Preliminaries on Continuous Logic

$\mathrm{lh}(\varphi)$ is the **length** of φ; i.e. count how many (not necessarily distinct) symbols appear in φ.

A wff $\varphi \in W_{\mathcal{L}}$ is **quantifier-free** if no quantifiers ($\sup_x$ or $\inf_x$) appear in φ.

A wff $\varphi \in W_{\mathcal{L}}$ is a **sentence** if all of its variable symbols are bound by quantifiers; i.e. its truth value is fixed only by the structure $\mathfrak{M}$.

Preliminaries on Continuous Logic

$\mathrm{lh}(\varphi)$ is the **length** of φ; i.e. count how many (not necessarily distinct) symbols appear in φ.

A wff $\varphi \in W_{\mathcal{L}}$ is **quantifier-free** if no quantifiers ($\sup_x$ or $\inf_x$) appear in φ.

A wff $\varphi \in W_{\mathcal{L}}$ is a **sentence** if all of its variable symbols are bound by quantifiers; i.e. its truth value is fixed only by the structure $\mathfrak{M}$.

The set of all sentences of $\mathcal{L}$ is denoted by $S_{\mathcal{L}}$.

Preliminaries on Continuous Logic

Define, for any sentence $\varphi \in S_{\mathcal{L}}$, the **value of** φ **in** $\mathfrak{M}$ in the natural way. Interpret all the symbols in the metric structure, and evaluate.

Preliminaries on Continuous Logic

Define, for any sentence $\varphi \in S_{\mathcal{L}}$, the **value of** φ **in** $\mathfrak{M}$ in the natural way. Interpret all the symbols in the metric structure, and evaluate.

Ex:

$$\left(\mathbf{d}(`\frac{1}{2}', `\frac{1}{4}')\right)^{\mathfrak{M}} = \left|\frac{1}{2} - \frac{1}{4}\right| = \frac{1}{4}$$

Preliminaries on Continuous Logic

Define, for any sentence $\varphi \in S_{\mathcal{L}}$, the **value of** φ **in** $\mathfrak{M}$ in the natural way. Interpret all the symbols in the metric structure, and evaluate.

Ex:

$$\left(\mathbf{d}\left(`\frac{1}{2}', `\frac{1}{4}'\right)\right)^{\mathfrak{M}} = \left|\frac{1}{2} - \frac{1}{4}\right| = \frac{1}{4}$$

For any $\varphi \in S_{\mathcal{L}}$, we say that $\mathfrak{M}$ **satisfies** φ, and write $\mathfrak{M} \models \varphi$, if $\varphi^{\mathfrak{M}} = 0$.

Preliminaries on Continuous Logic

At times it may be useful to consider the satisfiability of an arbitrary formula, rather than just sentences. Because of this, we may declare an **assignment** σ which assigns a unique point $a \in M$ to each variable $x \in V$. Satisfaction is defined naturally.

Preliminaries on Continuous Logic

Let $\mathcal{L}$ be a continuous signature with a metric.

(i) We call T a **theory** if T is a set of sentences of $\mathcal{L}$; i.e. $T \subseteq S_{\mathcal{L}}$.

Preliminaries on Continuous Logic

Let $\mathcal{L}$ be a continuous signature with a metric.

(i) We call T a **theory** if T is a set of sentences of $\mathcal{L}$; i.e. $T \subseteq S_{\mathcal{L}}$.

(ii) We call a theory T **complete** if there is a $\mathcal{L}$-structure $\mathfrak{M}$ s.t. for every sentence φ, we have $\varphi \in T$ if and only if $\mathfrak{M} \models \varphi$.

Preliminaries on Continuous Logic

We define the **degree of truth of φ with respect to T** as

$$\varphi_T^\circ := \sup\{\varphi^{\mathfrak{M}} : \mathfrak{M} \models T\}.$$

Preliminaries on Continuous Logic

We define the **degree of truth of φ with respect to T** as

$$\varphi_T^\circ := \sup\{\varphi^{\mathfrak{M}} : \mathfrak{M} \models T\}.$$

If T is complete, we say T is **decidable** if for every sentence φ, the value φ_T° is a computable real and, moreover, uniformly computable from φ.

Preliminaries on Continuous Logic

We define the **degree of truth of φ with respect to T** as

$$\varphi^{\circ}_{T} := \sup\{\varphi^{\mathfrak{M}} : \mathfrak{M} \models T\}.$$

If T is complete, we say T is **decidable** if for every sentence φ, the value φ°_{T} is a computable real and, moreover, uniformly computable from φ.

What the heck do we mean by computable from φ?

Effectivizing Continuous Logic

Recall that a **name** of a real number r is a sequence of natural numbers $\{k_n\}_{n=0}^{\infty}$ which serve as codes of a sequence of rational numbers $\{q_n\}_{n=0}^{\infty}$ s.t. $q_n \to r$ and for every n, for all $m > n$, $|q_n - q_m| < 2^{-(n+1)}$.

Effectivizing Continuous Logic

Recall that a **name** of a real number r is a sequence of natural numbers $\{k_n\}_{n=0}^{\infty}$ which serve as codes of a sequence of rational numbers $\{q_n\}_{n=0}^{\infty}$ s.t. $q_n \to r$ and for every n, for all $m > n$, $|q_n - q_m| < 2^{-(n+1)}$.

A real number r is **computable** if there is some effective procedure which outputs a name for r; i.e., r is computable if there is a c.e. name for r.

Effectivizing Continuous Logic

Let $\mathcal{L}$ be a signature.

(i) Define **bounded subtraction** $\dot{-} : [0,1]^2 \to [0,1]$ as

$$x \dot{-} y := \begin{cases} x - y & \text{if } y \leqslant x \\ 0 & \text{otherwise.} \end{cases}$$

Effectivizing Continuous Logic

Let $\mathcal{L}$ be a signature.

(i) Define **bounded subtraction** $\dot{-} : [0,1]^2 \to [0,1]$ as

$$x \dot{-} y := \begin{cases} x - y & \text{if } y \leqslant x \\ 0 & \text{otherwise.} \end{cases}$$

(ii) Define **negation** $\neg : [0,1] \to [0,1]$ as

$$\neg x := 1 - x.$$

Effectivizing Continuous Logic

(iii) Define the **half operator** $\frac{1}{2} : [0, 1] \to [0, 1]$ as

$$\frac{1}{2}\, x := \frac{x}{2}.$$

Effectivizing Continuous Logic

(iii) Define the **half operator** $\frac{1}{2} : [0,1] \to [0,1]$ as

$$\frac{1}{2}\, x := \frac{x}{2}.$$

Note that all three of the above functions are continuous from $[0,1]^n \to [0,1]$. Hence for every $\varphi, \psi \in W_{\mathcal{L}}$, we have $\varphi \dot{-} \psi \in W_{\mathcal{L}}$, $\neg\varphi \in W_{\mathcal{L}}$, and $\frac{1}{2}\, \varphi \in W_{\mathcal{L}}$.

Effectivizing Continuous Logic

Theorem

(Corollary 1.6, Ben Yaacov and Usvyatsov, 2010) The set of functions $\{\dot{-}, \neg, \frac{1}{2}\}$ *generates, for every* $n \in \mathbb{N}$*, a dense subset* $Ł^n$ *of the set of all continuous functions* $C[0,1]^n$ *in the uniform convergence topology.*

Effectivizing Continuous Logic

When referring to wffs, we will normally drop the parentheses around $\neg$, $\dot{-}$, and $\frac{1}{2}$ as indicative of the symbols which represent them.

Effectivizing Continuous Logic

When referring to wffs, we will normally drop the parentheses around $\neg$, $\dot{-}$, and $\frac{1}{2}$ as indicative of the symbols which represent them.

The set of **Ł-formulas** of $\mathcal{L}$, denoted $\text{Ł}_{\mathcal{L}}$ is the set of all wff's which only include $\neg$, $\dot{-}$, and $\frac{1}{2}$ instead of arbitrary continuous connectives.

Effectivizing Continuous Logic

When referring to wffs, we will normally drop the parentheses around $\neg$, $\dot{-}$, and $\frac{1}{2}$ as indicative of the symbols which represent them.

The set of **Ł-formulas** of $\mathcal{L}$, denoted Ł$_{\mathcal{L}}$ is the set of all wff's which only include $\neg$, $\dot{-}$, and $\frac{1}{2}$ instead of arbitrary continuous connectives. Ł-sentences and Ł-theories relativize.

Effectivizing Continuous Logic

When referring to wffs, we will normally drop the parentheses around $\neg$, $\dot{-}$, and $\frac{1}{2}$ as indicative of the symbols which represent them.

The set of **Ł-formulas** of $\mathcal{L}$, denoted $Ł_{\mathcal{L}}$ is the set of all wff's which only include $\neg$, $\dot{-}$, and $\frac{1}{2}$ instead of arbitrary continuous connectives. Ł-sentences and Ł-theories relativize.

N.B. while $|W_{\mathcal{L}}| = 2^{\aleph_0}$, we have that $|Ł_{\mathcal{L}}| = \aleph_0$. This collapse to countability will be necessary for executing effective procedures on the set of formulas.

Effectivizing Continuous Logic

We can define an **Ł-topology** on $W_{\mathcal{L}}$.

Effectivizing Continuous Logic

We can define an **Ł-topology** on $W_{\mathcal{L}}$.

Thus, given $\varphi \in W_{\mathcal{L}}$, call a sequence of Ł-neighborhoods $\{L_n\}_{n=0}^{\infty}$ which converge to φ an **Ł-name** of φ.

Effectivizing Continuous Logic

We can define an **Ł-topology** on $W_{\mathcal{L}}$.

Thus, given $\varphi \in W_{\mathcal{L}}$, call a sequence of Ł-neighborhoods $\{L_n\}_{n=0}^{\infty}$ which converge to φ an **Ł-name** of φ.

Corollary

For every $\varphi \in W_{\mathcal{L}}$, there exists an Ł-name of φ.

Effectivizing Continuous Logic

Let $\varphi \in W_{\mathcal{L}}$.

(i) By a **name** of φ we mean a sequence of natural numbers $\{k_n\}_{n=0}^{\infty}$ which serve as codes of a sequence of Ł-neighborhoods $\{L_n\}_{n=0}^{\infty}$ which form an Ł-name of φ.

Effectivizing Continuous Logic

Let $\varphi \in W_{\mathcal{L}}$.

(i) By a **name** of φ we mean a sequence of natural numbers $\{k_n\}_{n=0}^{\infty}$ which serve as codes of a sequence of Ł-neighborhoods $\{L_n\}_{n=0}^{\infty}$ which form an Ł-name of φ.

(ii) We say that φ is **computable** if there is some effective procedure which outputs a name for φ; i.e. if there exists some c.e. name for φ.

Effectivizing Continuous Logic

Let $\varphi \in W_{\mathcal{L}}$.

(i) By a **name** of φ we mean a sequence of natural numbers $\{k_n\}_{n=0}^{\infty}$ which serve as codes of a sequence of Ł-neighborhoods $\{L_n\}_{n=0}^{\infty}$ which form an Ł-name of φ.

(ii) We say that φ is **computable** if there is some effective procedure which outputs a name for φ; i.e. if there exists some c.e. name for φ.

(iii) Let T be a theory. We say that the map $\varphi \mapsto \varphi_T^{\circ}$ is **computable from** φ if there is some effective procedure which, given a name for φ, outputs a name for φ_T°.

Effectivizing Continuous Logic

A **computable presentation** of a metric structure $\mathfrak{M}$ is a pair $\mathfrak{M}^+ = (\mathfrak{M}, g)$ s.t. g maps $\mathbb{N}$ to some "dense" subset of M in a computable fashion, and all the functions and predicates of $\mathfrak{M}$ can be computed on the range of g.

Effectivizing Continuous Logic

A **computable presentation** of a metric structure $\mathfrak{M}$ is a pair $\mathfrak{M}^+ = (\mathfrak{M}, g)$ s.t. g maps $\mathbb{N}$ to some "dense" subset of M in a computable fashion, and all the functions and predicates of $\mathfrak{M}$ can be computed on the range of g.

All the points in the range of g are called **rational points** of $\mathfrak{M}^+$, and computable notions relativize.

Classical Result

Theorem (Effective Completeness of (Classical) First-Order Logic)

(Millar, 1978, appears in Harizanov) Let $\mathcal{L}$ be a first-order language and and T be a decidable $\mathcal{L}$-theory. Then there is a decidable model $\mathfrak{M}$ s.t. $\mathfrak{M} \models T$.

Classical Result

Theorem (Effective Completeness of (Classical) First-Order Logic)

(Millar, 1978, appears in Harizanov) Let $\mathcal{L}$ be a first-order language and and T be a decidable $\mathcal{L}$-theory. Then there is a decidable model $\mathfrak{M}$ s.t. $\mathfrak{M} \models T$.

This (more-or-less) translates to the continuous case.

Previous Result

Theorem

(Calvert, 2011) Let T be a complete, decidable continuous first-order theory. Then there is a probabilistically decidable continuous weak structure $\mathfrak{M}$ s.t. $\mathfrak{M} \models T$.

Previous Result

Theorem

(Calvert, 2011) Let T be a complete, decidable continuous first-order theory. Then there is a probabilistically decidable continuous weak structure $\mathfrak{M}$ s.t. $\mathfrak{M} \models T$.

Now let us consider the same problem in terms of computable presentations.

Effective Completeness

Let $\Gamma \subseteq W_{\mathcal{L}}$ and $\mathcal{D}$ be the set of dyadic numbers, i.e. numbers of the form $\frac{k}{2^n} \leqslant 1$ for $k, n \in \mathbb{N}_0$.

Effective Completeness

Let $\Gamma \subseteq W_{\mathcal{L}}$ and $\mathcal{D}$ be the set of dyadic numbers, i.e. numbers of the form $\frac{k}{2^n} \leqslant 1$ for $k, n \in \mathbb{N}_0$.

We say that Γ is **Henkin complete** if for every $\varphi \in Ł_{\mathcal{L}}$, every variable symbol x, and every pair $(p, q) \in \mathcal{D}$ s.t. $p < q$, there is some constant c s.t.

$$(\sup_x \varphi \dot{-} q) \wedge (p \dot{-} \varphi[c/x]) \in \Gamma.$$

Effective Completeness

Let $\Gamma \subseteq W_{\mathcal{L}}$ and $\mathcal{D}$ be the set of dyadic numbers, i.e. numbers of the form $\frac{k}{2^n} \leqslant 1$ for $k, n \in \mathbb{N}_0$.

We say that Γ is **Henkin complete** if for every $\varphi \in \text{Ł}_{\mathcal{L}}$, every variable symbol x, and every pair $(p, q) \in \mathcal{D}$ s.t. $p < q$, there is some constant c s.t.

$$(\sup_x \varphi \dot{-} q) \wedge (p \dot{-} \varphi[c/x]) \in \Gamma.$$

If we can extend T to a maximally consistent and Henkin complete set, terms built up from these constants can form the rational points of a computable presentation of a metric space $\mathfrak{M}$ which models T.

Effective Completeness

Lemma

Any complete, decidable theory T can be extended to a maximally consistent and Henkin complete set Λ where there is a uniformly computable sequence $\{\Lambda_s\}_{s\in\omega}$ of finite sets s.t. $\Lambda = \bigcup_{S\in\omega}\bigcap_{S\leqslant s<\omega}\Lambda_s$.

Effective Completeness

Lemma

Any complete, decidable theory T can be extended to a maximally consistent and Henkin complete set Λ where there is a uniformly computable sequence $\{\Lambda_s\}_{s\in\omega}$ of finite sets s.t. $\Lambda = \bigcup_{S\in\omega}\bigcap_{S\leqslant s<\omega}\Lambda_s$.

Proof sketch: Throw in a bunch of constants at every stage so all the Henkin business works nicely.

Effective Completeness

Lemma

Any complete, decidable theory T can be extended to a maximally consistent and Henkin complete set Λ where there is a uniformly computable sequence $\{\Lambda_s\}_{s\in\omega}$ of finite sets s.t. $\Lambda = \bigcup_{S\in\omega} \bigcap_{S\leqslant s<\omega} \Lambda_s$.

Proof sketch: Throw in a bunch of constants at every stage so all the Henkin business works nicely. At each stage s check all wff's φ of length less than s for relative consistency up to an error term of 2^{-s}.

Effective Completeness

Lemma

Any complete, decidable theory T can be extended to a maximally consistent and Henkin complete set Λ where there is a uniformly computable sequence $\{\Lambda_s\}_{s\in\omega}$ of finite sets s.t. $\Lambda = \bigcup_{S\in\omega} \bigcap_{S\leqslant s<\omega} \Lambda_s$.

Proof sketch: Throw in a bunch of constants at every stage so all the Henkin business works nicely. At each stage s check all wff's φ of length less than s for relative consistency up to an error term of 2^{-s}. For each of these, put them into Λ_s.

Effective Completeness

Lemma

Any complete, decidable theory T can be extended to a maximally consistent and Henkin complete set Λ where there is a uniformly computable sequence $\{\Lambda_s\}_{s\in\omega}$ of finite sets s.t.
$\Lambda = \bigcup_{S\in\omega} \bigcap_{S\leqslant s<\omega} \Lambda_s$.

Proof sketch: Throw in a bunch of constants at every stage so all the Henkin business works nicely. At each stage s check all wff's φ of length less than s for relative consistency up to an error term of 2^{-s}. For each of these, put them into Λ_s. These will all be effective (actually each is finite), but in the limit converge to a maximally consistent, Henkin complete extension.

Effective Completeness

Theorem (Effective Completeness of Continuous First-Order Logic)

Let $\mathcal{L}$ be a signature and T be a complete, decidable $\mathcal{L}$-theory. Then there is a computably presentable metric structure $\mathfrak{M}$ s.t. $\mathfrak{M} \models T$.

Effective Completeness

Proof sketch. Extend T to a Henkin complete and maximal consistent set Λ as outlined in the previous lemma.

Effective Completeness

Proof sketch. Extend T to a Henkin complete and maximal consistent set Λ as outlined in the previous lemma.

Let $\mathfrak{M}^*$ be the model of T whose universe M^* is the set of closed terms in C^* (the Henkin constants), and interpret functions, predicates, and terms as themselves.

Effective Completeness

Proof sketch. Extend T to a Henkin complete and maximal consistent set Λ as outlined in the previous lemma.

Let $\mathfrak{M}^*$ be the model of T whose universe M^* is the set of closed terms in C^* (the Henkin constants), and interpret functions, predicates, and terms as themselves.

Then $\mathfrak{M}^* \models T$.

Effective Completeness

Proof sketch. Extend T to a Henkin complete and maximal consistent set Λ as outlined in the previous lemma.

Let $\mathfrak{M}^*$ be the model of T whose universe M^* is the set of closed terms in C^* (the Henkin constants), and interpret functions, predicates, and terms as themselves.

Then $\mathfrak{M}^* \models T$. Further we can quotient out by the equivalence relation $t_0 \sim t_1$ iff $d(t_0, t_1) = 0$ to get a metric structure $\mathfrak{M}$ s.t. $\mathfrak{M} \models T$.

Effective Completeness

Now let g be any map which witnesses the computability of M^*, and define the presentation $\mathfrak{M}^+ := (\mathfrak{M}, g)$.

Effective Completeness

Now let g be any map which witnesses the computability of M^*, and define the presentation $\mathfrak{M}^+ := (\mathfrak{M}, g)$.

This presentation is computable since the rational points of M^+ are precisely $(M^*/\sim)$ and we can compute the value of $P^{\mathfrak{M}^*}(t_1, ..., t_{n(P)})$ up to arbitrary precision, by checking a large enough Λ_n from the previous proposition. '■'

Computational vs. Syntactic Complexity

Let $\mathfrak{M}$ be a computably presentable $\mathcal{L}$-structure computable presentation $\mathfrak{M}^\sharp$. We define the **atomic diagram** of $\mathfrak{M}$, denoted $\mathfrak{D}_{\mathfrak{M}}$, to be the set of atomic Ł-sentences or negated atomic Ł-sentences with only constant symbols interpreted as computable points of $\mathfrak{M}^\sharp$, which are satisfied by $\mathfrak{M}$.

Computational vs. Syntactic Complexity

Let $\mathfrak{M}$ be a computably presentable $\mathcal{L}$-structure computable presentation $\mathfrak{M}^\sharp$. We define the **atomic diagram** of $\mathfrak{M}$, denoted $\mathfrak{D}_{\mathfrak{M}}$, to be the set of atomic Ł-sentences or negated atomic Ł-sentences with only constant symbols interpreted as computable points of $\mathfrak{M}^\sharp$, which are satisfied by $\mathfrak{M}$.

Proposition

The atomic diagram of a computably presentable model is Π^0_1.

Computational vs. Syntactic Complexity

Proof. Let $\{\theta_k\}_{k=1}^{\infty}$ be an effective enumeration of the atomic and negated atomic Ł-sentences of $\mathcal{L}$ which only include the constant symbols interpreted as computable points of $\mathfrak{M}^\sharp$. Notice

$$\theta_s \in \mathfrak{D}_{\mathfrak{M}} \iff \forall n \in \mathbb{N},\ \mathfrak{M} \models \theta_s \dot{-} 2^{-n}. \blacksquare$$

Classical "Result"

Definition

(Harizanov, 1998) A (classical) model $\mathfrak{M}$ is computable if M is computable and some form of its atomic diagram is decidable.

Classical “Result”

Definition

(Harizanov, 1998) A (classical) model $\mathfrak{M}$ is computable if M is computable and some form of its atomic diagram is decidable.

Does this translate to the continuous case?

Classical "Result"

Definition

(Harizanov, 1998) A (classical) model $\mathfrak{M}$ is computable if M is computable and some form of its atomic diagram is decidable.

Does this translate to the continuous case? Not necessarily! Computably presentable structures can have non-computable atomic diagrams!

Computational vs. Syntactic Complexity

Lemma

There exists a uniformly computable sequence of real numbers $\{c_n\}_{n=0}^{\infty}$ *s.t.* $c_n \leqslant \frac{1}{2}$ *if and only if* $n \in K^c$.

Computational vs. Syntactic Complexity

Theorem

There exists a computably presentable model $\mathfrak{M}$ s.t. its atomic diagram $\mathfrak{D}_{\mathfrak{M}}$ is Π^0_1-complete.

Computational vs. Syntactic Complexity

Proof sketch. Let $\mathfrak{M} = ([0,1], |\cdot|, P_{\mathbb{Q}} : [0,1] \to [0,1], \mathbb{Q} \cap [0,1])$, where $P_{\mathbb{Q}} : [0,1] \to [0,1]$ denotes rational polynomials with domain $[0,1]$ and range bounded by $[0,1]$. Notice that this is a metric structure on the bounded metric space $([0,1], |\cdot|)$.

Computational vs. Syntactic Complexity

Proof sketch. Let $\mathfrak{M} = ([0,1], |\cdot|, P_{\mathbb{Q}} : [0,1] \to [0,1], \mathbb{Q} \cap [0,1])$, where $P_{\mathbb{Q}} : [0,1] \to [0,1]$ denotes rational polynomials with domain $[0,1]$ and range bounded by $[0,1]$. Notice that this is a metric structure on the bounded metric space $([0,1], |\cdot|)$.

Now fix $\{c_n\}_{n=0}^{\infty}$ a uniformly computable sequence of real numbers s.t. $c_n \leqslant \frac{1}{2}$ if and only if $n \in K^c$, which exists by the previous lemma, and throw them all in as as rational points for a computable presentation $\mathfrak{M}^\sharp$ of $\mathfrak{M}$.

Computational vs. Syntactic Complexity

We can now check the atomic diagram of $\mathfrak{M}$ for

$$\begin{aligned} c_n \leqslant \frac{1}{2} &\iff |c_n| \dot{-} \frac{1}{2} = 0 \\ &\iff \left|c_n \dot{-} \frac{1}{2}\right| = 0 \\ &\iff \big(d(0, f(\mathbf{c}_n))\big)^{\mathfrak{M}} = 0 \\ &\iff \mathfrak{M} \models d(0, f(\mathbf{c}_n)) \end{aligned}$$

where f is the function symbol corresponding to the bounded rational polynomial $x \dot{-} \frac{1}{2}$.

Computational vs. Syntactic Complexity

We can now check the atomic diagram of $\mathfrak{M}$ for

$$\begin{aligned} c_n \leqslant \frac{1}{2} &\iff |c_n| \dot{-} \frac{1}{2} = 0 \\ &\iff \left|c_n \dot{-} \frac{1}{2}\right| = 0 \\ &\iff \big(d(0, f(\mathbf{c}_n))\big)^{\mathfrak{M}} = 0 \\ &\iff \mathfrak{M} \models d(0, f(\mathbf{c}_n)) \end{aligned}$$

where f is the function symbol corresponding to the bounded rational polynomial $x \dot{-} \frac{1}{2}$. This allows us to decode K^c. '■'

Computational vs. Syntactic Complexity

Let $\mathcal{L}$ be a signature.

(i) We define the sets $\Sigma_0^0(\mathcal{L}) = \Pi_0^0(\mathcal{L}) = \Delta_0^0(\mathcal{L})$ as the set of all quantifier-free Ł-formulas of $\mathcal{L}$;

Computational vs. Syntactic Complexity

Let $\mathcal{L}$ be a signature.

(i) We define the sets $\Sigma_0^0(\mathcal{L}) = \Pi_0^0(\mathcal{L}) = \Delta_0^0(\mathcal{L})$ as the set of all quantifier-free Ł-formulas of $\mathcal{L}$;

(ii) for every $n \in \mathbb{N}$, we define $\Sigma_n^0(\mathcal{L})$ to be the set of all Ł-formulas of the form $\inf_{\overline{x}} \varphi$ s.t. $\overline{x}$ is a tuple of variable symbols and $\varphi \in \Pi_{n-1}^0$;

Computational vs. Syntactic Complexity

Let $\mathcal{L}$ be a signature.

(i) We define the sets $\Sigma^0_0(\mathcal{L}) = \Pi^0_0(\mathcal{L}) = \Delta^0_0(\mathcal{L})$ as the set of all quantifier-free Ł-formulas of $\mathcal{L}$;

(ii) for every $n \in \mathbb{N}$, we define $\Sigma^0_n(\mathcal{L})$ to be the set of all Ł-formulas of the form $\inf_{\overline{x}} \varphi$ s.t. $\overline{x}$ is a tuple of variable symbols and $\varphi \in \Pi^0_{n-1}$;

(iii) for every $n \in \mathbb{N}$, we define $\Pi^0_n(\mathcal{L})$ to be the set of all Ł-formulas of the form $\sup_{\overline{x}} \varphi$ s.t. $\overline{x}$ is a tuple of variable symbols and $\varphi \in \Sigma^0_{n-1}$.

Computational vs. Syntactic Complexity

Let $\mathcal{L}$ be a signature.

(i) We define the sets $\Sigma_0^0(\mathcal{L}) = \Pi_0^0(\mathcal{L}) = \Delta_0^0(\mathcal{L})$ as the set of all quantifier-free Ł-formulas of $\mathcal{L}$;

(ii) for every $n \in \mathbb{N}$, we define $\Sigma_n^0(\mathcal{L})$ to be the set of all Ł-formulas of the form $\inf_{\overline{x}} \varphi$ s.t. $\overline{x}$ is a tuple of variable symbols and $\varphi \in \Pi_{n-1}^0$;

(iii) for every $n \in \mathbb{N}$, we define $\Pi_n^0(\mathcal{L})$ to be the set of all Ł-formulas of the form $\sup_{\overline{x}} \varphi$ s.t. $\overline{x}$ is a tuple of variable symbols and $\varphi \in \Sigma_{n-1}^0$.

$\Sigma_n^0(S_{\mathcal{L}})$, $\Pi_n^0(S_{\mathcal{L}})$, and $\Delta_n^0(S_{\mathcal{L}})$ relativize as sets of Ł-sentences of $\mathcal{L}$ in the standard way.

Computational vs. Syntactic Complexity

Theorem

(Camrud, Goldbring, McNicholl) Given a computably presentable $\mathcal{L}$-structure $\mathfrak{M}$ with computable presentation $\mathfrak{M}^\sharp$,

Computational vs. Syntactic Complexity

Theorem

(Camrud, Goldbring, McNicholl) Given a computably presentable $\mathcal{L}$-structure $\mathfrak{M}$ with computable presentation $\mathfrak{M}^\sharp$,

$$\{(\varphi, r) \ : \ \varphi \in \Delta^0_0(S_{\mathcal{L}}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^{\mathfrak{M}} \leqslant r\} \ \in \ \Pi^0_1$$

Computational vs. Syntactic Complexity

Theorem

(Camrud, Goldbring, McNicholl) Given a computably presentable $\mathcal{L}$-structure $\mathfrak{M}$ with computable presentation $\mathfrak{M}^\sharp$,

$$\{(\varphi, r) \ : \ \varphi \in \Delta_0^0(S_{\mathcal{L}}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^{\mathfrak{M}} \leqslant r\} \ \in \ \Pi_1^0$$

$$\{(\varphi, r) \ : \ \varphi \in \Delta_0^0(S_{\mathcal{L}}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^{\mathfrak{M}} < r\} \ \in \ \Sigma_1^0$$

Computational vs. Syntactic Complexity

For each $n \in \mathbb{N}$,

$$\{(\varphi, r) \ : \ \varphi \in \Pi^0_n(S_{\mathcal{L}}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^{\mathfrak{M}} \leqslant r\} \ \in \ \Pi^0_n$$

Computational vs. Syntactic Complexity

For each $n \in \mathbb{N}$,

$$\{(\varphi, r) \ : \ \varphi \in \Pi^0_n(S_{\mathcal{L}}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^{\mathfrak{M}} \leqslant r\} \ \in \ \Pi^0_n$$

and

$$\{(\varphi, r) \ : \ \varphi \in \Pi^0_n(S_{\mathcal{L}}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^{\mathfrak{M}} < r\} \ \in \ \Sigma^0_{n+1}$$

Computational vs. Syntactic Complexity

For each $n \in \mathbb{N}$,

$$\{(\varphi, r) \ : \ \varphi \in \Sigma_n^0(S_{\mathcal{L}}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^{\mathfrak{M}} \leqslant r\} \ \in \ \Pi_{n+1}^0$$

Computational vs. Syntactic Complexity

For each $n \in \mathbb{N}$,

$$\{(\varphi, r) \ : \ \varphi \in \Sigma^0_n(S_\mathcal{L}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^\mathfrak{M} \leqslant r\} \ \in \ \Pi^0_{n+1}$$

and

$$\{(\varphi, r) \ : \ \varphi \in \Sigma^0_n(S_\mathcal{L}) \ \wedge \ r \in \mathcal{D} \ \wedge \ \varphi^\mathfrak{M} < r\} \ \in \ \Sigma^0_n$$

Computational vs. Syntactic Complexity

Proof sketch. *(i)* Notice that we may effectively enumerate all $\varphi \in \Delta_0^0(S_{\mathcal{L}})$.

Computational vs. Syntactic Complexity

Proof sketch. (i) Notice that we may effectively enumerate all $\varphi \in \Delta_0^0(S_{\mathcal{L}})$. Then effectively enumerate all possible pairs

$$\{(\varphi_s, r_s) \ : \ \varphi_s \in \Delta_0^0(S_{\mathcal{L}}) \ \wedge \ r_s \in \mathcal{D}\}_{s=1}^{\infty}.$$

Computational vs. Syntactic Complexity

Proof sketch. (i) Notice that we may effectively enumerate all $\varphi \in \Delta_0^0(S_\mathcal{L})$. Then effectively enumerate all possible pairs

$$\{(\varphi_s, r_s) \ : \ \varphi_s \in \Delta_0^0(S_\mathcal{L}) \ \wedge \ r_s \in \mathcal{D}\}_{s=1}^{\infty}.$$

At stage s, recall that since $\mathfrak{M}^\sharp$ is computable and φ_s is quantifier-free containing only constant symbols interpreted as computable points of $\mathfrak{M}^\sharp$, we can compute a Cauchy name for $\varphi_s^{\mathfrak{M}}$; call it $\{q_m\}_{m=1}^{\infty}$.

Computational vs. Syntactic Complexity

Notice that

$$\begin{aligned} \varphi_s^{\mathfrak{M}} \leqslant r_s &\iff \forall m \in \mathbb{N},\ \varphi_s^{\mathfrak{M}} < r_s + 2^{-m} \\ &\iff \forall m \in \mathbb{N},\ q_m < r_s + 2^{-m}, \end{aligned}$$

with this condition being Π_1^0.

Computational vs. Syntactic Complexity

Notice that

$$\varphi_s^{\mathfrak{M}} \leqslant r_s \iff \forall m \in \mathbb{N},\ \varphi_s^{\mathfrak{M}} < r_s + 2^{-m}$$
$$\iff \forall m \in \mathbb{N},\ q_m < r_s + 2^{-m},$$

with this condition being Π_1^0.Notice also

$$\varphi_s^{\mathfrak{M}} < r_s \iff \exists n \in \mathbb{N},\ \varphi_s^{\mathfrak{M}} < r_s - 2^{-n}$$
$$\iff \exists m, n \in \mathbb{N},\ q_m < r_s - 2^{-n},$$

with this condition being Σ_1^0. ■ *(i)*

Computational vs. Syntactic Complexity

Notice that

$$\varphi_s^{\mathfrak{M}} \leqslant r_s \iff \forall m \in \mathbb{N},\ \varphi_s^{\mathfrak{M}} < r_s + 2^{-m}$$
$$\iff \forall m \in \mathbb{N},\ q_m < r_s + 2^{-m},$$

with this condition being Π_1^0.Notice also

$$\varphi_s^{\mathfrak{M}} < r_s \iff \exists n \in \mathbb{N},\ \varphi_s^{\mathfrak{M}} < r_s - 2^{-n}$$
$$\iff \exists m, n \in \mathbb{N},\ q_m < r_s - 2^{-n},$$

with this condition being Σ_1^0. ■ *(i)*

(ii) and (iii) relativize similarly. '■'

Moving Forward

(i) Generalize optimal bound results to further diagrams (upper bounds already proven).

Moving Forward

(i) Generalize optimal bound results to further diagrams (upper bounds already proven).

(ii) Extend complexity results to infinitary formulas.

Bibliography

Itai Ben Yaacov, Alexander Berenstein, C. Ward Henson, and Alexander Usvyatsov, "Model theory for metric structures," *Model Theory with Applications to Algebra and Analysis*, vol. 2, (Zo´e Chatzidakis, Dugald Macpherson, Anand Pillay, and Alex Wilkie, editors) London Math Society Lecture Note Series, vol. 350, London, 2008, pp. 315-427.

Itai Ben Yaacov and Arthur Paul Pedersen, "A proof of completeness for continuous first-order logic," *The Journal of Symbolic Logic*, vol. 75 (2010), no. 1, pp. 168-190.

Itai Ben Yaacov and Alexander Usvyatsov, "Continuous First Order Logic and Local Stability," *Transaction of the American Mathematical Society*, vol. 362, no. 10, 2010, pp. 5213-5259.

Bibliography

Klaus Weihrauch, *Computable Analysis: An Introduction*, Texts in Theoretical Computer Science. An EATCS Series, Springer, 2000.

V. S. Harizanov, "Pure computable model theory," *Handbook of Recursive Mathematics*, vol. 1, (Yu. L. Ershov, S. S. Goncharov, A. Nerode, J.B. Remmel, and V. W. Marek, editors) Studies in Logic and the Foundations of Mathematics, 138-139, North-Holland, Amsterdam, 1998, pp. 3-114.

Wesley Calvert, "Metric structures and probabilistic computation," *Theoretical Computer Science*, vol. 412 (2011), pp. 2766-2775.

Questions?

Feel free to reach out to me at

ccamrud@iastate.edu

Or comment on this video!