VON NEUMANN, CONNES, TURING

JOEL B. NEWMAN

September 23, 2023

Abstract. The Connes Embedding Problem (CEP) had been an open problem in the theory of von Neumann algebras since the 1970's when, in 2020, it was resolved by the finding that $\mathsf{MIP}^* = \mathsf{RE}$, a computational complexity result involving quantum entanglement. Thus, the study of the CEP and its resolution brings together the fields of Functional Analysis, Computational Complexity Theory, Mathematical Quantum Physics, and—with the alternative connection between the two results found by Goldbring and Hart—Model Theory.

Over the summer of 2023, I had the opportunity to engage in an independent reading project on this topic in order to better understand these fields and the result, overseen and mentored by Prof. Marcin Sabok and Prof. Prakash Panangaden. The following is my write-up of this project, which attempts to explain the CEP and its resolution by $\mathsf{MIP}^* = \mathsf{RE}$ through Model Theory at an elementary level, borrowing heavily from Goldbring's excellent Guided Tour paper [1].

Contents

1. Operator Algebras 1
1.1. $*$- and C*-algebras 2
1.2. von Neumann Algebras 3
1.3. The Connes Embedding Problem 6
2. Model Theory 9
2.1. Classical Model Theory 9
2.2. Continuous Model Theory 14
3. Computational Complexity 17
3.1. Modeling Computing 17
3.2. Interactive Proofs and Games 18
3.3. $\mathsf{MIP}^* = \mathsf{RE}$ 21
4. From CEP To Computability 22
4.1. From CEP to Model Theory 22
4.2. Computability 22
5. From MIP* To Non-Computability 24
References 25

1. Operator Algebras

It all starts with operator algebras. Given a Hilbert space $\mathcal{H}$, we call any continuous (or equivalently, bounded) linear function $\mathcal{H} \to \mathcal{H}$ an operator.[1] We denote the space of all such operators on $\mathcal{H}$ by $\mathcal{B}(\mathcal{H})$. It has a natural structure as an algebra over $\mathbb{C}$ (essentially just a vector space plus bilinear "multiplication," which in this case is defined by composition), whence we call $\mathcal{B}(\mathcal{H})$ and any subset closed

[1]Some sources call any function between topological vector spaces an operator, and refer instead to "bounded linear operators."

under these operators an operator algebra. Actually, we will be slightly stricter and require that any operator algebra contain its identity $\mathrm{Id}_{\mathcal{H}}$; other sources call such an algebra *unital*.

1.1. **$*$- and C*-algebras.** While the name "operator algebra" really just means that the algebra is made of operators, I think that it is very apt, as in the field of math studying operator algebras, we repeatedly find structures that may similatenously defined *concretely*, according to naturally properties of the operators that make it up, and *algebraically*, using only relations.

An elementary result in functional analysis gives that for any $T \in \mathcal{B}(\mathcal{H})$, there exists a $T^* \in \mathcal{B}(\mathcal{H})$ such that for all $\xi, \eta \in \mathcal{H}$, $\langle T\xi, \eta\rangle = \langle \xi, T^*\eta\rangle$, known as its Hermitian adjoint, or just adjoint. We can then canonize the $*$-operation, and say that any subalgebra of $\mathcal{B}(\mathcal{H})$ closed under $*$ is a $*$-subalgebra of $\mathcal{B}(\mathcal{H})$. But we may just as well define this algebraically: an $*$-algebra over $\mathbb{C}$ is an algebra $\mathcal{A}$ over $\mathbb{C}$ with associative multiplication and with an additional operation $* : \mathcal{A} \to \mathcal{A}$ satisfying that for all $x, y \in \mathcal{A}$, $\lambda \in \mathbb{C}$,

- $(x+y)^* = x^* + y^*$
- $(xy)^* = y^*x^*$
- $(x^*)^* = x$
- $(\lambda x)^* = \overline{\lambda}x^*$ (where $\overline{\lambda}$ is the complex conjugate of λ).
- $\mathrm{Id}^* = \mathrm{Id}$

Returning to elementary definitions from Functional Analysis, we may define a norm on $\mathcal{B}(\mathcal{H})$ by

$$\begin{aligned}\|T\| &:= \sup_{\xi \in \mathcal{H}} \frac{\|T\xi\|_{\mathcal{H}}}{\|\xi\|_{\mathcal{H}}} \\ &= \sup_{\substack{\xi \in X \\ \|\xi\|_{\mathcal{H}} = 1}} \|T\xi\|_{\mathcal{H}}.\end{aligned}$$

This norm is known as the operator norm. In fact, any given linear function $T : \mathcal{H} \to \mathcal{H}$, we have that $T \in \mathcal{B}(\mathcal{H})$ if and only if $\|T\|$ is finite, and that gives that T is bounded. With this, we are ready for our first proper subheaded definition:

Definition 1.1.1 (C*-algebra)**.** *Concretely*, we call a $*$-subalgebra of $\mathcal{B}(\mathcal{H})$ a C*-algebra if it is closed under the topology on $\mathcal{H}$ induced by the operator norm, which we call the *operator norm topology*.

Algebraically, we call a $*$-algebra $\mathcal{A}$ together with a norm $\|\cdot\|$ a C*-algebra if it satisfies the following relations for $x, y \in \mathcal{A}$:

- $\|xy\| \leq \|x\|\|y\|$
- $\|x^*\| = \|x\|$
- $\|x^*x\| = \|x\|^2$ *(The C* identity).*

The definitions coincide, such that any C*-subalgebra of $\mathcal{B}(\mathcal{H})$ is algebraically a C*-algebra, and any algebraic C*-algebra is isomorphic to a C*-subalgebra of $\mathcal{B}(\mathcal{H})$.

1.1.1. *Positives, Projections, Partial isometries, Oh My!.* There exists a taxonomy of various different kinds of elements of C*-algebras, most of which have very natural concrete and algebraic definitions. Let $\mathcal{A} \subseteq \mathcal{B}(\mathcal{H})$ be a C*-algebra.

- $x \in \mathcal{A}$ is self-adjoint if it has real spectrum – that is, $\sigma(x) = \{\lambda \mid x - \lambda \operatorname{Id}_{\mathcal{H}} \text{ has no inverse in } \mathcal{B}(\mathcal{H})\} \subseteq \mathbb{R}$. Equivalently, x is self-adjoint if $x = x^*$.
- $x \in \mathcal{A}$ is positive if for all $\xi \in \mathcal{H}$, $\langle x\xi, \xi\rangle \geq 0$; equivalently, x is positive if $x = y^*y$ for some $y \in \mathcal{A}$. (Note that this means that any positive element is self-adjoint.)
- $x \in \mathcal{A}$ is a projection if it projects vectors onto some closed subset of $\mathcal{H}$; equivalently, x is a projection if it is both self-adjoint and idempotent (i.e. $x^* = x = x^2$.)
- $x \in \mathcal{A}$ is unitary or a unitary if it is surjective and preserves the inner product (i.e. for all $\xi, \eta \in \mathcal{H}$, $\langle x\xi, x\eta\rangle = \langle \xi, \eta\rangle$); equivalently, x is a unitary if $x^*x = xx^* = \operatorname{Id}_{\mathcal{H}}$.
- $x \in \mathcal{A}$ is a partial isometry if it is an isometry (between $\mathcal{H}$ and $\mathcal{H}$) on $(\ker x)^\perp$, the orthogonal compliment of its kernel; equivalently, x is a partial isometry if x^*x is a projection.

With these definitions, we may introduce a partial order on $\mathcal{B}(\mathcal{H})$: for $a, b \in \mathcal{A}$, we say that $a \leq b$ if $b - a$ is positive.

On projections in specific, we can see that for p, q projections in $\mathcal{A}$, $p \leq q$ if and only if $p\mathcal{H} \subseteq q\mathcal{H}$, and so $\leq$ forms a total order on the set of all projections. More than that, it actually forms an *ortholattice* – that is, a complemented, bounded lattice:

$$\begin{aligned} p \curlywedge q &:= \overset{\text{strong}}{\lim_{n\to\infty}} (pq)^n \\ &= \text{orthogonal projection onto } p\mathcal{H} \cap q\mathcal{H} \\ p^\perp &:= \operatorname{Id}_{\mathcal{H}} - p \\ p \curlyvee q &:= (p^\perp \curlywedge q^\perp)^\perp \\ &= \text{orthogonal projection onto } \operatorname{Cl}(p\mathcal{H} + q\mathcal{H}). \end{aligned}$$

Moreover, this lattice is *complete* in the sense that every subset (including the empty set $\emptyset$) has a join and a meet.

1.2. **von Neumann Algebras.** In Definition 1.1.1, we referred to the operator norm topology in order to define a C*-algebra as a $*$-algebra which is closed in this topology. We now introduce two new (weaker) topologies: the strong and weak operator topology.

We define the strong operator topology (SOT) as the topology of pointwise convergence; that is, for a sequence $(T_n)_{n\in\mathbb{N}}$ of elements of $\mathcal{B}(\mathcal{H})$, $T_n \to T$ in the SOT if and only if for each $\xi \in \mathcal{H}$, $\|T_n\xi - T\xi\| \to 0$. Slightly less straightforward is the weak operator topology (WOT), which we define as the weakest topology on $\mathcal{B}(\mathcal{H})$ such that for all $\xi, \eta \in \mathcal{H}$, the map $T \mapsto \langle T\xi, \eta\rangle$ is continuous.

We also introduce a new algebraic concept: for any subalgebra $\mathcal{A} \subseteq \mathcal{B}(\mathcal{H})$, we define its commutant $\mathcal{A}'$ to be the set of all elements of $\mathcal{B}(\mathcal{H})$ that commute with every element of $\mathcal{A}$; that is, $\mathcal{A}' \coloneqq \{b \in \mathcal{B}(\mathcal{H}) \mid \forall a \in \mathcal{A} : ab = ba\}$. We then define the bicommutant $\mathcal{A}''$ in the obvious way: $\mathcal{A}'' = (\mathcal{A}')'$.

With these, we may now define von Neumann Algebras:

Definition 1.2.1 (von Neumann algebra)**.** *Concretely*, a von Neumann Algebra is a $*$-subalgebra of $\mathcal{B}(\mathcal{H})$ which is closed in either the weak or the strong operator topology; as it so happens, these are equivalent.

Algebraically,[2] a von Neumann algebra is a $*$-subalgebra $\mathcal{A}$ of $\mathcal{B}(\mathcal{H})$ which is equal to its own bicommutant: $\mathcal{A} = \mathcal{A}''$. Of course, this definition is also equivalent to the others.

1.2.1. *A New Order.* We now introduce a relation on projections in a von Neumann algebra, and a corresponding equivalence relation:

Definition 1.2.2 (Murray-von Neumann (sub-)equivalence)**.** Let $\mathcal{M}$ be a von Neumann algebra. For projections $p, q \in \mathcal{M}$, we say that p is (Murray-von Neumann) sub-equivalent to q (denoted $p \preccurlyeq q$) if there exists a partial isometry $u \in \mathcal{M}$ with $uu^* = p$ and $u^*u \leq q$.

We say that p and q are (Murray-von Neumann) equivalent (denoted $p \approx q$) if there exists a partial isometry $u \in \mathcal{M}$ such that $uu^* = p$ and $u^*u = q$.

Now, $\preccurlyeq$ is not (in general) a partial order on the projections of a von Neumann algebra; rather, it is a partial order on the $\approx$*-equivalence classes* of projections.

We call a projection $p \in \mathcal{M}$ infinite if there exists $q \in \mathcal{M}$ with $q < p$ (i.e. a proper subprojection) such that $p \approx q$.

1.2.2. *Factors.* Let the center of a von Neumann algebra $\mathcal{M}$ be defined as

$$\mathcal{Z}(\mathcal{M}) \coloneqq \{x \in \mathcal{M} \mid \forall a \in \mathcal{M} : xa = ax\}.$$

With center, projections, and $\preccurlyeq$ all defined, we can finally get to the purpose of this discussion:

Definition 1.2.3 (factor)**.** We call a von Neumann algebra $\mathcal{M}$ a factor if $\mathcal{Z}(\mathcal{M}) = \mathbb{C} \cdot \mathrm{Id}_{\mathcal{H}}$.

While we term this a "factor", we are not very interested here in factorizing von Neumann algebras.[3] Rather, we are interested in a taxonomy of factors containing a very important type – a taxonomy enabled by the following property of factors:

Theorem 1.2.4 ([2] Theorem 6.1.8)**.** *If $\mathcal{M}$ is a factor, then $\preccurlyeq$ is a total order on $\approx$-equivalence classes.*

With this defined, we can now discuss the complete classification of factors. In [2], Sir Vaughan Jones gives a very elegant system for classifying factors using $\preccurlyeq$

[2]In constrast to our discussion so far, von Neumann algebras are firmly a "concrete" notion – they are not defined as universal algebras as in the abstract definition for Cstar-algebras in Definition 1.1.1.

[3]Indeed, Jones refers to said notion of factorization as "bizarre".

that will will repeat here. The idea is that Theorem 1.2.4 allows us to classify factors by the $\preccurlyeq$-order type of their $\approx$-equivalence classes This gives us a classification as follows:

Factor Type	**$\preccurlyeq$-Order Type**	**Is $\mathrm{Id}_{\mathcal{H}}$ Infinite?**
Type I_n	$\{0, 1, 2, ..., n\}$	No
Type I_∞	$\{0, 1, 2, ..., \infty\}$	Yes
Type II_1	$[0, 1]$	No
Type II_∞	$[0, \infty]$	Yes
Type III	$\{0, \infty\}$	Yes

TABLE 1. Classification of Factors by $\preccurlyeq$-Order Type.

Note that in Table 1, we include ∞ in the order type only when $\mathrm{Id}_{\mathcal{H}}$ is infinite in the sense discussed at the end of Section 1.2.1; strictly speaking, $[0, 1]$ and $[0, \infty]$ are order isomorphic, and so are $\{0, 1\}$ and $\{0, \infty\}$. In the case that $\mathrm{Id}_{\mathcal{H}}$ is infinite, we call the entire factor infinite.

1.2.3. *Traces and II_1 Factors.* Having been introduced to II_1 factors, we now take another pass at the concept from another angle. A functional is a continuous/ bounded linear function from a C^* algebra $\mathcal{A}$ to $\mathbb{C}$. We define a trace as follows:

Definition 1.2.5 (state, trace)**.** Given a C*-algebra $\mathcal{A}$, we call a functional ϕ…

- positive if $\phi(a^*a) \geq 0$ for all $a \in \mathcal{A}$ (i.e. it maps positive operators to positive scalars).
- faithful if $\phi(a^*a) = 0 \Rightarrow a = 0$ for $a \in \mathcal{A}$.
- a state if ϕ is positive and $\phi(\mathrm{Id}_{\mathcal{H}}) = 1$.
- normal if, for $\mathcal{A}_1$ the operator norm unit ball of $\mathcal{A}$, $\phi \restriction_{\mathcal{A}_1} : \mathcal{A}_1 \to \mathbb{C}$ is continuous.
- tracial if for $a, b \in \mathcal{A}$, $\phi(ab) = \phi(ba)$.

We call a normal, faithful, tracial state a trace.

Comparing Definition 1.2.5 to Definition 1.2.2 with Theorem 1.2.4 in mind, we begin to see a clear relationship between the trace and Murray-von Neumann subequivalence, particularly in a factor $\mathcal{M}$ that admits a trace:

In this case, for two projections $p, q \in \mathcal{M}$,

$$\begin{aligned} p \approx q \Longrightarrow \mathrm{tr}(p) &= \mathrm{tr}(uu^*) \\ &= \mathrm{tr}(u^*u) \\ &= \mathrm{tr}(q) \end{aligned}$$

where u is some partial isometry in $\mathcal{M}$, and

$$\begin{aligned} p \preccurlyeq q \Longrightarrow \mathrm{tr}(p) &= \mathrm{tr}(uu^*) \\ &= \mathrm{tr}(u^*u) \\ &\leq \mathrm{tr}(q) \end{aligned}$$

where u is, again, some partial isometry in $\mathcal{M}$.

Noting that the preceeding discussion also implies that an infinite factor may not admit a trace, we have thus almost completely justified the equivalence to this next definition, our second pass at defining type II_1 factors:

Definition 1.2.6 (type II_1 factor)**.** We call any infinite dimensional factor that admits a trace a type II_1 factor.

This definition is equivalent to the definition given in Table 1.

Note that in a II_1 factor, the trace is in fact unique, subject to $\mathrm{tr}(\mathrm{Id}_{\mathcal{H}}) = 1$.

1.3. **The Connes Embedding Problem.** We now know what a II_1 factor is, but there is still some groundwork to cover to get to the Connes Embedding Problem in earnest.

First, we define embedding. An embedding of tracial von Neumann algebras (i.e. von Neumann algebras and their traces) is a normal, injective $*$-homomorphism that preserves the trace.

Our next task is to construct a very particular II_1 factor.

1.3.1. *Constructing the Hyperfinite II_1 Factor.* We call a separable von Neumann algebra $\mathcal{M}$ hyperfinite if it contains an increasing union of finite-dimensional subalgebras whose union is WOT-dense; that is, there exists some $(\mathcal{A}_n)_n$ such that $\mathcal{A}_n \subseteq \mathcal{A}_{n+1}$ and such that the weak closure of $\bigcup_n \mathcal{A}_n$ is $\mathcal{M}$.

We will take this definition as a hint in our quest to assemble the largest II_1 factor possible.

Definition 1.3.1 ($\mathcal{R}$, the hyperfinite II_1 factor)**.** Let $M_\infty(\mathbb{C})$ denote the matrix algebra on $\mathbb{C}$ of infinite-dimensional matrices. For any $n \geq 1$, we may embed $M_{2^n}(\mathbb{C}) \hookrightarrow M_\infty(\mathbb{C})$ diagonally as block matrices:

$$\begin{aligned} M_{2^n}(\mathbb{C}) &\hookrightarrow M_\infty(\mathbb{C}) \\ A &\mapsto \begin{pmatrix} A & 0 & \cdots \\ 0 & A & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix} \\ &= A \otimes I_\infty \end{aligned}$$

where I_∞ is the identity matrix in $M_\infty(\mathbb{C})$ and $\otimes$ is the Kronecker product.

Note that under this embedding, we have that for each $n \geq 1$, $M_{2^n}(\mathbb{C}) \subseteq M_{2^{n+1}}(\mathbb{C})$ via the diagonal embedding:

$$\begin{aligned} M_{2^n}(\mathbb{C}) &\hookrightarrow M_{2^{n+1}}(\mathbb{C}) \hookrightarrow M_\infty(\mathbb{C}) \\ A \mapsto \begin{pmatrix} A & 0 \\ 0 & A \end{pmatrix} &\mapsto \begin{pmatrix} A & 0 & 0 & 0 & \cdots \\ 0 & A & 0 & 0 & \cdots \\ 0 & 0 & A & 0 & \cdots \\ 0 & 0 & 0 & A & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \\ = A \otimes I_2 \qquad &= A \otimes I_2 \otimes I_\infty = A \otimes I_\infty. \end{aligned}$$

Let $\mathcal{A}_\infty = \bigcup_n M_{2^n}(\mathbb{C}) \subseteq M_\infty(\mathbb{C})$ according to the embedding given above.

Each $M_{2^n}(\mathbb{C})$ admits a matrix trace (i.e. the sum of its diagonal elements), which we will denote by Tr_{2^n}. Note that this trace is preserved through embeddings between these matrix algebras, so we may define a trace $\mathrm{tr} : \mathcal{A}_\infty \to \mathbb{C}$ that is normalized (in the sense that the identity has a trace of 1) by

$$\begin{aligned} \mathrm{tr} : \mathcal{A}_\infty &\to \mathbb{C} \\ A \in M_{2^n}(\mathbb{C}) &\mapsto \frac{1}{n} \mathrm{Tr}_{2^n}(A). \end{aligned}$$

We then use this to define an inner product $\langle \cdot, \cdot \rangle_{\mathcal{A}_\infty}$ on $\mathcal{A}_\infty$, by

$$\langle A, B \rangle_{\mathcal{A}_\infty} := \mathrm{tr}(B^* A).$$

This allows us to take the *Cauchy completion* of $\mathcal{A}_\infty$, upgrading $\mathcal{H}$ from an inner product space to a Hilbert space $\mathcal{H}$.

Now, we consider $\mathcal{B}(\mathcal{H})$. As $\mathcal{A}_\infty$ is dense in $\mathcal{H}$, we can uniquely determine a representation of $\mathcal{A}_\infty$ as a subset of $\mathcal{B}(\mathcal{H})$ by

$$\begin{aligned} \pi : \mathcal{A}_\infty &\to \mathcal{B}(\mathcal{H}) \\ A &\mapsto \pi(A) \end{aligned}$$

with

$$\begin{aligned} \pi(\xi) : \mathcal{H} &\to \mathcal{H} \\ \eta &\mapsto \xi\eta \end{aligned}$$

identifying $\mathcal{A}_\infty \subseteq \mathcal{H}$. Through the following representation, we can find $\pi(\mathcal{A}_\infty) \subseteq \mathcal{B}(\mathcal{H})$. Finally, by taking

$$\mathcal{R} := \pi(\mathcal{A}_\infty)'' = \overline{\pi(\mathcal{A}_\infty)}^{\text{strong}} = \overline{\pi(\mathcal{A}_\infty)}^{\text{weak}}$$

we obtain $\mathcal{R}$, which we call the hyperfinite II_1 factor. We identify $\mathcal{A}_\infty$ as a subalgebra of $\mathcal{R}$.

We can see easily that $M_n(\mathbb{C})$ is always a factor; as such we can be sure that $A_\infty \cap \mathcal{Z}(\mathcal{R}) = \mathbb{C} \cdot \mathrm{Id}_{\mathcal{H}}$. By checking that the completion does not add any extra non-$\mathbb{C} \cdot \mathrm{Id}_{\mathcal{H}}$ elements to the center $\mathcal{Z}(\mathcal{R})$, we can then be sure that $\mathcal{R}$ is a factor. It is also easy enough to see that $\mathcal{R}$ is infinite dimensional, and we have clearly defined it to admit a trace, so $\mathcal{R}$ is in fact a II_1 factor.

By Theorem 15.1.1 in [2] (originally proven by Murray and von Neumann themselves), there is only one hyperfinite II_1 factor up to isomorphism. Moreover, it can be shown that $\mathcal{R}$ embeds into any II_1 factor.

1.3.2. *Ultrapowers (of von Neumann algebras).* For our next step, we will need to define ultrafilters.

Definition 1.3.2 (ultrafilter)**.** Given an arbitrary set X, an ultrafilter on X is a set $\mathfrak{U} \subseteq \mathcal{P}(X)$ that satisfies:

1. $\emptyset \notin \mathfrak{U}$ (*nontriviality*)
2. $X \subseteq B \subseteq A \in \mathfrak{U} \implies B \in \mathfrak{U}$ (*upwards closure*)
3. $A, B \in \mathfrak{U} \implies A \cap B \in \mathfrak{U}$ (*downwards direction*)
4. $A \in \mathfrak{U}$ or $A^\complement \in \mathfrak{U}$ for all $A \in \mathcal{P}(X)$ (*completeness*)

We may alternatively define an ultrafilter $\mathfrak{U}$ on X as a $\{0,1\}$-valued probability measure on $\mathfrak{U}$, identifiying the measure as $\mathfrak{U}(S) = 1$ when $S \in \mathfrak{U}$ for $S \subseteq X$.

The probability measure definition suggests a natural interpretation of $\mathfrak{U}$ as a probability measure in which every subset S of X is either $\mathfrak{U}$-*big* and occurs $\mathfrak{U}$-*almost always* or is $\mathfrak{U}$-*small* and occurs $\mathfrak{U}$-*almost never*. This notion is instructive for our next definition:

Definition 1.3.3 (ultralimit)**.** Let $(\xi_i)_{i\in I}$ be a bounded sequence of complex numbers. Then, for any ultrafilter $\mathfrak{U}$, there exists a unique $\xi' \in \mathbb{C}$ such that for all $\varepsilon > 0$, $\{i : |\xi' - \xi_i| < \varepsilon\} \in \mathfrak{U}$. In other words, for all $\varepsilon > 0$, ξ_i is ε-close to ξ' for $\mathfrak{U}$-almost all $i \in I$. We call this the $\mathfrak{U}$-ultralimit of the sequence, and denote it by $\xi' = \lim_{i\in I}^{\mathfrak{U}} x_i$.

Ultrafilters may either be *principal* or *free*. For an element $a \in X$, the principal ultrafilter of a on X is the ultrafilter $\mathfrak{U}_a = \{S \subseteq X : a \in S\}$. If an ultrafilter is principal of any element, we call if principal, and otherwise call it free. Note that for a sequence $(\xi_i)_{i\in I}$ as above, $\lim_i^{\mathfrak{U}_{i_0}} = \xi_{i_0}$. On the other hand, for a free $\mathfrak{U}$, $\lim_i^{\mathfrak{U}} \xi_i$ will be an accumulation point of $(\xi_i)_{i\in I}$.

We now perform the *tracial ultraproduct construction* on $(\mathcal{R}, \mathrm{tr})$:

Definition 1.3.4 ($\mathcal{R}^\omega/\mathfrak{U}$)**.** Fix some free ultrafilter $\mathfrak{U}$, the choice of which is irrelevant. Consider $\mathcal{R}^\omega = \prod_{i\in\omega} \mathcal{R}$, the Cartesian product consisting of all ω-indexed sequences in $\mathcal{R}$. Then, define

$$\ell^\infty(\mathcal{R}) := \left\{ x \in \mathcal{R}^\omega : \sup_{i\in\omega} \|x(i)\| < \infty \right\}$$

where $\|\cdot\|$ is the operator norm. If we then define a on $\ell^\infty(\mathcal{R})$ by

$$\|x\|_{\ell^\infty(\mathcal{R})} := \sup_{i\in\omega} \|x(i)\|$$

we can show that $\ell^\infty(\mathcal{R})$ together with $\|\cdot\|_{\ell^\infty(\mathcal{R})}$) is a C*-algebra.

Now, we want to define a trace on $\ell^\infty(\mathcal{R})$. Unfortunately, $\mathrm{tr}(x) := \lim_i^{\mathfrak{U}} \mathrm{tr}(x(i))$ does not work, as we may have that positive elements of $\ell^\infty(\mathcal{R})$ (i.e., $x \in \ell^\infty(\mathcal{R})$ such that $x(i)$ is positive for each $i \in \omega$) can be such that $\lim_i^{\mathcal{R}} \mathrm{tr}(x(i)) = 0$. As such, we must define

$$N_{\mathfrak{U}} := \left\{ x \in \ell^\infty(\mathcal{R}) : \lim_i^{\mathfrak{U}} \mathrm{tr}(x(i)) = 0 \right\}$$

and quotient by this (after verifying that $N_{\mathfrak{U}}$ is a two-sided ideal in $\ell^\infty(\mathcal{R})$). This gives us $\ell^\infty(\mathcal{R})/N_{\mathfrak{U}}$, which which we denote by $\mathcal{R}^\omega/\mathfrak{U}$, and which happens to be a von Neumann algebra with trace

$$\mathrm{tr}\Big([x]_{N_{\mathfrak{U}}}\Big) := \lim_{i\in\omega}^{\mathfrak{U}} \mathrm{tr}(x(i)).$$

1.3.3. *The Statement.* We can now finally state the Connes embedding problem:

Definition 1.3.5 (Connes Embedding Problem)**.** The Connes Embedding Problem (CEP) states that every tracial separable von Neumann algebra embeds into some ultrapower of the unique hyperfinite II_1 factor $\mathcal{R}$.

It can be shown (see [1], p.519) that every tracial von Neumann algebra embeds into a II_1 factor, via a procedure that enlarges the tracial von Neumann algebra into a II_1 factor. Hence, we may take the CEP to be the statement that II_1 factor embeds into $\mathcal{R}^\kappa/\mathfrak{U}$ for some κ and $\mathfrak{U}$.

2. Model Theory

In general, the field of mathematics consists of choosing some mathematical object, and then studying it by writing down provable facts. Model Theory formalizes this fact, by fixing a formal language for constructing sentences and formualas, and then studying the relationship between this and our *model*, the object itself.

In the following section, we borrow liberally from the presentation of Model Theory given in [3].

2.1. Classical Model Theory.

In the following, we will notate $\bar{x}$ for a sequence of symbols or variables $x_0, x_1, ..., x_n$ of arbitrary length.

2.1.1. *Constants and Functions.* Formally, to study a type of mathematical object using Model Theory, we first fix a (first-order) language L, which symbols we can use to construct various types of utterances; these utterances are given meaning through models or L-structures, which give interpretations and allow our utterances to describe actual mathematical objects, or be either true or false.

First-order languages L may contain constant symbols c, each of which gets interpreted in an L-structure A as an element $c^A \in A$, and function symbols f, each of which gets interpreted as a function $f^A : A \to A$. We can use these language's, as well as the background logic's variable symbols $x_0, x_1, ...$, to construct L-terms. Recursively applying these interpretations to an L-term τ that does not contain variable symbols allows it to be interpreted as $\tau^A \in A$, an element of A.

We can do much the same with terms that *do* contain variables by providing a substitution, although the formal procedure is a little complicated. Model Theory in general must always name an object to use it, so if we want to substitute the variables $\bar{x}$ in L-term $\tau(\bar{x})$ with elements $\bar{a} \in A$, we first name them, assigning each x_i a constant symbol c_i; we denote the expanded language we have created by adding these symbols $L(\bar{c})$. We then expand A into an $L(\bar{c})$-structure that interprets each c_i as a_i, and denote the resulting structure $(A, \bar{a})$. In this way, we can interpret $[\tau(\bar{a})]^A = [\tau(\bar{c})]^{(A,\bar{a})}$.

In the other direction to expanding the language, when L' is a larger language than L we may consider an L'-structure A' as an L-structure by "forgetting" the extra symbols, notating the resulting structure by $A' \restriction L$.

When a subset B of (the universe of) L-structure A is such that $c^A \in B$ for all constant symbols $c \in L$ and $f^A(\bar{b}) \in B$ for all function symbols $f \in L$ and $\bar{b} \in B$, then it is impossible that an L-term using interpretations from A and parameters in B refers to an element outside of B, and so we can regard B as another L-structure with the same interpretations as A but a smaller universe; in this case we call B a substructure of A, and notate it $A \leq B$.

2.1.2. *Relations and Formulae.* L may also contain relation symbols R, which are interpreted in an L-structure A as $R^A \subseteq A^n$ — (n-ary) relations on A. We may construct an atomic L-formula using a relation symbol applied to L-terms, or using the background logic's $=$ relation, which has a fixed interpretation as the relation $\{(a,a) : a \in A\}$. When an atomic L-formula φ contains no variables, then after interpreting it in A we either have a true or false statement; if φ is true in A we say A models φ, and notate this $A \vDash \varphi$.

We may construct more complex quantifier-free L-formulae φ by combining other quantifier-free L-formulae with logical connectives, which we interpret in the natural way. Like for terms, when an L-formula has a variable we must *bind* it in order to fully interpret the formula, but we also have the option of using quantifiers.

Quantifier symbols also come from the background logic, and they work like this: the formula $(\forall x)\varphi(x)$ interpreted in A is true when $\varphi(a)$ is true for all $a \in A$, and $(\exists x)\varphi(x)$ is true in A when there exists $a \in A$ such that $A \vDash \varphi(a)$. A (general) L-formula is built up from quantifier-free L-formulae only (and optionally) by adding quantifiers. An L-formula in which every variable is bound is called an L-sentence. We also call a formula that uses only $\forall$ universal and a formula that uses only $\exists$ existential.

2.1.3. *Theories & Elementary Equivalence.* When Φ is a set of L-sentences and A an L-structure such that $A \vDash \varphi$ for every $\varphi \in \Phi$, we say that $A \vDash \Phi$, and in general extend most notation for formulas to sets of formulas ($\Phi(\bar{x})$ for a set of formulas with free variables in $\bar{x}$, etc). If for all models A such that $A \vDash \Phi$ we have that $A \vDash \psi$, we say that Φ entails ψ, notated $\Phi \vdash \psi$.

When A, B are L-structures, we will notate that for all $\varphi \in \Phi$, $A \vDash \varphi \Longrightarrow B \vDash \varphi$ by

$$A \overset{\Phi}{\Rrightarrow} B.$$

If $A \Rrightarrow^{\Phi} B$ and $B \Rrightarrow^{\Phi} A$, then we notate this by $A \equiv^{\Phi} B$.

We may define the theory of L as the set of all L-sentences, and denote this $\mathrm{Th}(L)$. Note that as $\mathrm{Th}(L)$ is closed under negation, if $A \Rrightarrow^{\mathrm{Th}(L)} B$ then we must also have $A \Lleftarrow^{\mathrm{Th}(L)} B$; in this case we notate this simply by $A \equiv B$ and call the structures elementarily equivalent.

We may also consider the universal theory of L ($\mathrm{Th}_{\forall}(L)$) which consists of all universal L-sentences, and the existential theory $\mathrm{Th}_{\forall}(L)$, consisting of all existential sentences. Note that when $A \leq B$, then we have that

$$A \overset{\mathrm{Th}_{\exists}(L)}{\Rrightarrow} B$$

as if there exists a satisfactory element in A it must also exist in B, and

$$A \overset{\mathrm{Th}_{\forall}(L)}{\Lleftarrow} B.$$

as if all elements of B are satisfactory all elements of A are as well.

The converse does not precisely hold, but a similar statement does:

Theorem 2.1.1 ([3], Corollary 5.4.3)**.** *If A and B are such that*

$$A \overset{Th_{\forall}(L)}{\Lleftarrow} B$$

then there is a structure $A' \geq A$ such that $A' \equiv B$.

Note the above notation is nontraditional; traditionally we would use $\mathrm{Th}(A)$ to notate the set of all L-sentences φ such that $A \vDash \varphi$, and likewise for $\mathrm{Th}_\forall(A)$ and $\mathrm{Th}_\exists(B)$. For example, instead of saying $A \Rrightarrow^{\mathrm{Th}_\forall(L)} B$, we'd say that $B \vDash \mathrm{Th}_\forall(A)$. We use the $\Rrightarrow$ notation so that when we get to continuous logic in Section 2.2, our notation remains compatible.

2.1.4. *Elementary Substructures & Embeddings.* We say that an L-structure A is an elementary substructure of a substructure B (notated $A \preccurlyeq B$) if for all $\bar{a} \in A$,

$$(A, \bar{a}) \overset{\mathrm{Th}(L,\bar{c})}{\equiv} (B, \bar{a}).$$

Confusingly, this is a much stronger notion than either the property of being a substructure or that of elementary equivalence; $A \leq B$ and $A \equiv B$ does not imply $A \preccurlyeq B$, although the converse does hold.

We also introduce the concept of an embedding:

Definition 2.1.2 (embedding)**.** For A and B L-structures, we call an injection $f : A \to B$ an embedding if it satisfies

1. $f\big(c^A\big) = c^B$ for all constant symbols c.
2. $\bar{a} \in R^A \iff f\bar{a} \in R^B$ for all tuples $\bar{a}$ and compatible relation symbols R.
3. $f\big(g^A(\bar{a})\big) = g^B(f\bar{a})$ for all tuples $\bar{a}$ and compatible function symbols g.

When $f : A \to B$ is an embedding, then we may refer to the substructure fA of B.

We tie the utility of these concepts together with the following proposition:

Theorem 2.1.3 (Elementary Diagram Lemma; [3] Lemma 2.5.3)**.** *Let $\bar{a}$ generate A, in the sense that any element of a can be referred to by an L-term with parameters in $\bar{a}$.*

Then, if is B such that

$$(A, \bar{a}) \overset{Th(L(\bar{c}))}{\Leftarrow} B$$

there is an embedding $f : A \to B \restriction L$ such that fA is an elementary substructure of $B \restriction L$.

We refer to such an embedding as an elementary embedding.

2.1.5. *Types.* Let A be an L-structure, and X a subset of (the universe of) A. Then, for some tuple $\bar{b}$ of elements of A, the complete type of $\bar{b}$ over X with respect to A consists of the set of all L-formulas $\varphi(\bar{x}, \bar{y})$ such that $A \vDash \varphi\big(\bar{b}, X\big)$. Intuitively, this consists of everything we can say in language L about $\bar{b}$ using X that is true in A. We notate this by $\mathrm{tp}_A\big(\bar{b}/X\big)$. More generally, for $p(\bar{x})$ a set of formulas in the variables $\bar{x}$, we say that $p(\bar{x})$ is a complete type of $\bar{b}$ over X if it is a complete type of $\bar{b}$ over X *with respect to* B, where B is some elementary extension of A.

We then call any subset of a complete type over X (with respect to A) a type over X (with respect to A). We say that a type $\Phi(\bar{x})$ is realized by a tuple $\bar{b}$ in A if $\Phi \subseteq \mathrm{tp}_A\big(\bar{b}/X\big)$, and if Φ is not realized by any tuple in A we say that A omits Φ.

With this, with state without proof a very important theorem about types:

Theorem 2.1.4 ([3], Theorem 5.2.1)**.** *For $\Phi(\bar{x})$ a set of L-formulas with parameters from $X \subseteq A$, where A is an L-structure, $\Phi(\bar{x})$ is a type over X with respect to A if and only if Φ is finitely realized in A. Moreover, $\Phi(\bar{x})$ is a* complete *type over X with respect to A if and only if Φ is* maximal *(with respect to inclusion) with the property that it is finitely realized in A.*

2.1.6. *Saturation and Ultrapowers.* With the concept of types under our belt, we can introduce λ-saturated L-structures. Informally, a λ-saturated L-structure is one that is "large" enough to contain any element that we can describe using fewer than λ of its elements. Formally:

Definition 2.1.5 (λ-saturated L-structure)**.** We call a L-structure A λ-saturated if for all subsets $X \subseteq A$ with $|X| < \lambda$, all complete types $\Phi(x)$ over X with respect to A are realized by elements of A.

If an L-structure A is $|A|$-saturated, we simply call it saturated.

How do we obtain such a structure? As it turns out, using *another* (recall Section 1.3.2) thing called an ultrapower!

Definition 2.1.6 (ultrapower of an L-structure)**.** Let A be an L-structure, and I an index set. We will construct the ultrapower step-by-step.

First, we construct the *direct power* A^I as the set of all I-indexed sequences (which we will treat as maps $I \to A$). Letting $\varphi(\bar{x})$ be a L-formula and $(a_0, a_1, a_2, ...) = \bar{a}$ a tuple in A^I, we notate $\|\varphi(\bar{a})\| := \{i \in I : A \vDash \varphi(\bar{a}(i))\}$, where $\bar{a}(i) = (a_0(i), a_1(i), a_2(i), ...)$. We call this the boolean value of $\varphi(\bar{a})$.

Next, we take some ultrafilter $\mathfrak{U}$ (recall Definition 1.3.2) and define an equivalence relation $\overset{\mathfrak{U}}{\sim}$ on A^I by $a \overset{\mathfrak{U}}{\sim} b \iff \|a = b\| \in \mathfrak{U}$. We denote the equivalence class of $a \in A^I$ by $a/\mathfrak{U}$.

Finally, we define the ultrapower as

$$A^I/\mathfrak{U} := \{a/\mathfrak{U} : a \in A^I\}$$

where we interpret symbols as follows:

1. for a constant symbol c,
$$c^{A^I/\mathfrak{U}} = \big(c^{A_i} : i \in I\big)/\mathfrak{U}$$
2. for a function symbol f,
$$f^{A^I/\mathfrak{U}}(a_0/\mathfrak{U}, ..., a_{n-1}/\mathfrak{U}) = \big(f^{A_i}(\bar{a}(i)) : i \in I\big)/\mathfrak{U}$$
3. for a relation symbol R,
$$(a_0/\mathfrak{U}, ..., a_{n-1}/\mathfrak{U}) \in R^{A^I/\mathfrak{U}} \iff \|R(\bar{a})\| \in \mathfrak{U}.$$

Defining it in this way, we obtain that $A^I/\mathfrak{U} \vDash \varphi(\bar{a})$ if and only if $\|\varphi(\bar{a})\| \in \mathfrak{U}$, by a theorem of Łoś.

The resemblance to the construction in Section 1.3.2 is not in name only, and the analogy between the two will be further deepened in Section 2.2.

Regardless, if we choose the ultrafilter in the above definition precisely, we can have our desired saturated models:

Theorem 2.1.7 ([4] and [5] 10.5, as cited in [6] Theorems 4 and 5)**.** *Let L be a first-order language with $|L| \leq \kappa$ (that is, that names fewer than κ symbols in total). Let A be a L-structure. Then there exists an ultrafilter $\mathfrak{U}$ on κ such that the ultrapower $A^I/\mathfrak{U}$ is κ^+-saturated.*

2.1.7. *Frayne's Theorem.* We finish with the following theorem:

Theorem 2.1.8 (Frayne's Theorem; [3] Section 8.5, Exercise 8)**.** *Two L-structures A and B are elementarily equivalent if and only if there exists an embedding f such that fA is an elementary substructure of some ultrapower $B^I/\mathfrak{U}$.*

Proof. In the following, we will say that an L-formula $\varphi(\bar{x})$ is *absolute* between M and N where $M \leq N$ if for every $\bar{a} \in M$, we have that $M \vDash \varphi(\bar{a}) \iff N \vDash \varphi(\bar{a})$; letting $\bar{a}$ generate M, this is equivalent to saying that $(M, \bar{a}) \equiv^{\varphi(\bar{c})} (N, \bar{a})$.

Let I be some index set, $\mathfrak{U}$ an ultrafilter, and $f : A \to B^I/\mathfrak{U}$ an elementary embedding (i.e., an embedding such that $fA \preccurlyeq B$). Taking $\bar{a}$ to be the empty 0-tuple, by definition of elementary embedding we have that for all sentences ϕ of L,

$$A \vDash \phi \iff B^I/\mathfrak{U} \vDash \phi$$

and so clearly $A \equiv B^I/\mathfrak{U}$. Similarly, as the diagonal embedding $e : B \to B^I/\mathfrak{U}$ is an elementary embedding, we have that $B^I/\mathfrak{U} \equiv B$ and so $A \equiv B$.

Let A, B be L-structures such that $A \equiv B$. Choose some sequence $\bar{a}$ of A that generates A of length κ. Using Theorem 2.1.7, we choose some ultrafilter $\mathfrak{U}$ such that $B^\kappa/\mathfrak{U}$ is κ^+-saturated. I claim that there exists $\bar{d}$ in $B^\kappa/\mathfrak{U}$ of length κ such that

$$(A, \bar{a}) \equiv \left(B^\kappa/\mathfrak{U}, \bar{d}\right)$$

as $L(\bar{c})$-structures.

I will prove this by induction, and will show that for each $\iota \leq \kappa$,

$$(A, \bar{a} \upharpoonright \iota) \equiv \left(B^\kappa/\mathfrak{U}, \bar{d} \upharpoonright \iota\right)$$

as $L(\bar{c} \upharpoonright \iota)$-structures.

In the zero case, we have that $A \equiv B$ and we have previously established that $\mathrm{Th}(B) = \mathrm{Th}(B^\kappa/\mathfrak{U})$ using the diagonal embedding. As such, we have that $\mathrm{Th}(A) = \mathrm{Th}(B) = \mathrm{Th}(B^\kappa/\mathfrak{U})$ and so $A \equiv B^\kappa/\mathfrak{U}$.

In the step case, we assume that we have $(A, \bar{a} \upharpoonright \iota) \equiv \left(B^\kappa/\mathfrak{U}, \bar{d} \upharpoonright \iota\right)$. We then let $\Phi(\bar{x}, y)$ be the complete type of a_ι over $\bar{a} \upharpoonright \iota$, in the sense that for all L-formulas $\phi(\bar{x}, y)$ such that $A \vDash \phi(\bar{a} \upharpoonright \iota, a_\iota)$, $\phi \in \Phi$. Now, let Ψ be an arbitrary finite subset of Φ, and consider the L-formula

$$\exists y \bigwedge_{\psi \in \Psi} \psi(\bar{x} \upharpoonright \iota, y)$$

which we will denote as $\rho_\Psi(\bar{x})$. Let σ_Ψ be the $L(\bar{c} \upharpoonright \iota)$-sentence obtained by replacing each occurrence of x_η with the constant symbol c_η for $\eta < \iota$. We can clearly see that $A \vDash \rho_\Psi(\bar{a} \upharpoonright \iota)$ and thus that $(A, \bar{a} \upharpoonright \iota) \vDash \sigma_\Psi$ for each finite $\Psi \subseteq \Phi$, and so as $(A, \bar{a} \upharpoonright \iota) \equiv \left(B^\kappa/\mathfrak{U}, \bar{d} \upharpoonright \iota\right)$, we have that $\left(B^\kappa/\mathfrak{U}, \bar{d}\right) \vDash \sigma_\Psi$. We thus have that

$$B^\kappa/\mathfrak{U} \vDash \rho_\Psi\left(\bar{d} \upharpoonright \iota\right)$$

for each finite $\Psi \subseteq \Phi$. We thus have that $B^\kappa/\mathfrak{U}$ *finitely realizes* $\Phi(\bar{x}, y)$ over $\bar{d} \restriction \iota$ and thus that $\Phi(\bar{x}, y)$ is a type over $\bar{d} \restriction \iota$ with respect to $B^\kappa/\mathfrak{U}$ by Theorem 2.1.4. Now, as $B^\kappa/\mathfrak{U}$ is κ^+-saturated and $\iota \leq \kappa < \kappa^+$, we have that $B^\kappa/\mathfrak{U}$ realizes this type, and so there exists an element of $B^\kappa/\mathfrak{U}$ we denote by d_ι such that for all $\phi \in \Phi$, $B^\kappa/\mathfrak{U} \vDash \phi\big(\bar{d} \restriction \iota, d_\iota\big)$. We thus have that every $L(c \restriction \iota + 1)$-sentence is absolute between A and $B^\kappa/\mathfrak{U}$ and so $(A, \bar{a} \restriction \iota + 1) \equiv \big(B^\kappa/\mathfrak{U}, \bar{d} \restriction \iota + 1\big)$.

In the limit case, we let ϕ be an arbitrary $L(\bar{c} \restriction \lambda)$-sentence, where $\lambda \leq \kappa$ is a limit ordinal. As ϕ is finitely long, it uses only finitely many constant symbols, and so there exists a greatest $\iota < \lambda$ such that c_ι is in ϕ. But then ϕ is a $L(\bar{c} \restriction \iota + 1)$-sentence, and as we assume that $(A, \bar{a} \restriction \iota + 1) \equiv \big(B^\kappa/\mathfrak{U}, \bar{d} \restriction \iota + 1\big)$ by our induction hypothesis, we have that ϕ must be absolute between $(A, \bar{a} \restriction \iota + 1)$ and $\big(B^\kappa/\mathfrak{U}, \bar{d} \restriction \iota + 1\big)$.

We thus have completed the induction, and thus proven the claim. Immediately from Theorem 2.1.3 we obtain that as $\bar{a}$ generates A, that A is elementarily embeddable into $B^\kappa/\mathfrak{U}$. □

We use this to obtain a completely unmotivated corollary, whose importance will only become obvious in Section 4.1:

Corollary 2.1.9. *Let A and B be two L-structures. Then*

$$A \overset{Th_\forall(L)}{\Leftarrow} B$$

if and only if A is embeddable into an ultrapower of B.

Proof. First, assume that $A \Leftarrow^{\mathrm{Th}_\forall(L)} B$. Then Theorem 2.1.1 gives that A is a substructure of some L-structure A' such that $A' \equiv B$. Then, Theorem 2.1.8 gives that there is an elementary embedding $f : A \to B^I/\mathfrak{U}$ for some index set I and ultrafilter $\mathfrak{U}$.

We can then embed A into A' using the inclusion $i : A \to A'$, and so $fi : A \to B^I/\mathfrak{U}$ is an embedding of A into an ultrapower of B.

On the other hand, if $f : A \to B^I/\mathfrak{U}$ is an embedding, then we have that

$$A \overset{\mathrm{Th}_\forall(L)}{\Leftarrow} B^I/\mathfrak{U} \equiv B$$

and so

$$A \overset{\mathrm{Th}_\forall(L)}{\Leftarrow} B.$$

□

2.2. **Continuous Model Theory.** For studying C* and von Neumann algebras, we consider a kind of *continuous logic* and its corresponding *continuous Model Theory*. What this means is that rather than giving each well-formed sentence in our logic a truth value (i.e. evaluating it as either true or false), each attains some real number value when evaluated in a model.

2.2.1. *Goldbring and Hart's Continuous Logic.* We briefly review the continuous logic as presented in [1] for studying tracial von Neumann algebras.

Definition 2.2.1 (L_{vNa})**.** For a von Neumann algebra $\mathcal{M}$, let $\mathcal{M}_1$ denote the operator norm unit ball (i.e. $\mathcal{M}_1 \coloneqq \{x \in \mathcal{M} \mid \|x\| \leq 1\}$). We will refer to any expression built from indeterminates $x_1, x_2, ..., x_n$ using addition, multiplication, subtraction, multiplication, and $*$ as a $*$-polynomial.

- Let $\mathcal{F}$ denote the set of all $*$-polymomials $p(x_1, ..., x_n)$ with $n \geq 0$ such that for any von Neumann algebra $\mathcal{M}$, $p(\mathcal{M}_1^n) \subseteq \mathcal{M}_1$.
- Our formal language will be $L_{\text{vNa}} = F \cup \{\text{tr}_{\mathfrak{R}}, \text{tr}_{\mathfrak{I}}, d\}$, where $\text{tr}_{\mathfrak{R}}$ and $\text{tr}_{\mathfrak{I}}$ are the real and imaginary parts of the trace on $\mathcal{M}$, respectively, and d is the metric $d(a,b) = \|a-b\|_{\text{tr}} = \sqrt{\langle a-b, a-b\rangle_{\text{tr}}} = \text{tr}((a-b)^*(a-b))$.
- A basic L_{vNa}-formula will be of the form $\text{tr}_{\mathfrak{R}}(p(\bar{x}))$, $\text{tr}_{\mathfrak{I}}(p(\bar{x}))$, or $d(p(\bar{x}), q(\bar{x}))$ for $p, q \in \mathcal{F}$.
- A quantifier-free L_{vNa}-formula is of the form $f(\varphi_1, ..., \varphi_m)$ where $f : \mathbb{R}^m \to \mathbb{R}$ is a continuous function and $\varphi_1, ..., \varphi_m$ are basic L_{vNa}-formulae.
- A *general* L_{vNa}-formula will be of the form

$$\left(\underset{x_{i_1}}{Q^1}\right) \cdots \left(\underset{x_{i_k}}{Q^k}\right) \varphi(x_1, ..., x_n)$$

  where each $i_j \in \{1, ..., n\}$. $\varphi(x_1, ..., x_n)$ is a quantifier-free L_{vNa}-formula, and each Q^i is either sup or inf.

We call any L_{vNa}-formula in which there are no free variables a L_{vNa}-sentence.

For any L_{vNa}-formula $\varphi(x_1, ..., x_n)$, von Neumann algebra $\mathcal{M}$, and $\bar{a} \in \mathcal{M}_1^n$, we may evaluate $\varphi(\bar{a})^{\mathcal{M}}$ to be the real number obtained by replacing the variables $\bar{x}$ with the values $\bar{a}$; this real number valuation takes the place of the truth value that would be obtained in a classical logic.

2.2.2. *Relating Continuous and Classical Logic.* We will for the most part avoid working directly in continuous logic or our continuous model theory, for the simple reason that it is infernally finnicky and irritating to formalize. Thankfully, most results carry over directly, once we fix an analogy between the two systems.

The basic concept in this relationship is this, and I will emphasize it since it is counterintuitive: in continuous logic, we will in general replace the notion of $A \vDash \varphi$ with $\varphi^{\mathcal{M}} = 0$; that is we will take a sentence to be satisfied by a model when its evaluation in that model is zero – especially when the sentence has nonnegative range. In fact, we will use $\mathcal{M} \vDash \varphi$ and the like to notate that $\varphi^{\mathcal{M}} = 0$.

We then consider a L_{vNa}-sentence with positive range φ universal when it is "quantified" only by sup, and existential when quantified only by inf. This means that a universal L_{vNa}-sentence $\left(\sup_{x_{i_1}}\right) \cdots \left(\sup_{x_{i_n}}\right) \varphi(\bar{x})$ (where φ is quantifier-free) is satisfied only when $\varphi^{\mathcal{M}}(\bar{a}) = 0$ for all $\bar{a} \in \mathcal{M}$, and likewise for existential sentences.

We also will expand our $\Rrightarrow$ notation; we will use $\mathcal{M} \Rrightarrow^{\Phi} \mathcal{N}$ to notate that for all L_{vNa}-sentences φ, $\varphi^{\mathcal{M}} \geq \varphi^{\mathcal{N}}$. Note that through our analogy, this implies the same thing that $\Rrightarrow$ meant for classical model theory: if $\mathcal{M} \vDash \varphi$ and so $\varphi^{\mathcal{M}} = 0$, then $0 \leq \varphi^{\mathcal{N}} \leq \varphi^{\mathcal{M}} \leq 0$ and so $\varphi^{\mathcal{N}} = 0$ and $\mathcal{N} \vDash \varphi$.

Again, this notation is nontraditional; the usual notation is that $\mathrm{Th}(\mathcal{M})$ is a function from $\mathrm{Th}(L_{\mathrm{vNa}})$ (as we have defined it) to $\mathbb{R}$, and likewise for $\mathrm{Th}_\forall$ and $\mathrm{Th}_\exists$. We would then state something like

$$\mathcal{M} \overset{\mathrm{Th}_\forall(L_{\mathrm{vNa}})}{\Rrightarrow} \mathcal{N}$$

by $\mathrm{Th}_\forall(\mathcal{M}) \geq \mathrm{Th}_\forall(\mathcal{N})$, to mean that the former function is always greater than the latter one. We will prefer this notation to the $\Rrightarrow$ one for the purpose of saving space, but make sure to recall the connection between the two, as it allows us to interpret our classical model theory results in the continuous case.

Under this identification, *nearly all* of the results that we have stated in the previous section still hold. In particular, Frayne's Theorem (Theorem 2.1.8) and its corrolary (Corollary 2.1.9) which will be important later. Moreover, many operator-algebraic properties are identical to model-theoretic properties. It is easy to see that a L_{vNa}-substructure is the same as a $*$-subalgebra of a tracial von Neumann algebra that preserves that trace. Moreover, an embedding of tracial von Neumann algebras is injective, a $*$-homomorphism (and so preserves interpretations of $*$-polynomials), and is trace-preserving (and so preserves interpretations of $\mathrm{tr}_{\mathfrak{R}}$, $\mathrm{tr}_{\mathfrak{I}}$, and d); thus, it is also an embedding of L_{vNa}-structures.[4] Of course, not all results remain — for example $A \Rrightarrow^{\mathrm{Th}(L)} B$ implies $A \equiv B$ for classical L-structures but $\mathcal{M} \Rrightarrow^{\mathrm{Th}(L)} \mathcal{N}$ does <u>not</u> imply the same for tracial von Neumann algebras, it instead implies $\mathrm{Th}(A) \geq \mathrm{Th}(B)$ — so some care must be taken, but many do.

2.2.3. *Ultrapowers.* Which brings us to ultrapowers. We have seen two different things called ultrapowers in this paper: the ultrapower of the hyperfinite II_1 factor (which we will heretofore notate as $\mathcal{R}^{\mathfrak{U}}$), given in Definition 1.3.4, and ultrapowers of models, given above in Definition 2.1.6. The former starts by taking $\mathcal{R}^\omega$, the ω -indexed sequences of $\mathcal{R}$, prunes it to bounded sequences with $\ell^\infty(\mathcal{R})$, and then reduces it by quotienting out $N_{\mathfrak{U}}$, the elements of $\ell^\infty(\mathcal{R})$ that would have zero trace under the natural trace definition we obtained with $\mathfrak{U}$-ultralimits. The latter starts by taking A^I, the I-indexed sequences of A, then quotients it by the $\overset{\mathfrak{U}}{\sim}$ equivalence relation, which says that $a \overset{\mathfrak{U}}{\sim} b$ if $\|a = b\| \in \mathfrak{U}$.

As it turns out, under our analogy between continuous and classical model theory (subject to a little bookkeeping), these processes are identical; we take the ultrapower $\mathcal{M}^{\mathfrak{U}}$ in the continuous model theory sense in the same way we took the ultrapower $\mathcal{R}^{\mathfrak{U}}$.

2.2.4. *Soundness and Completeness.* The last thing that we note is that there is a *sound* and *complete* proof system for L_{vNa}. Recalling the concept of entailment, in classical logic for a language L, a proof system would be a set of rules that can be used to generate "proofs" of finite length for statements of the form $\Psi \vdash \varphi$, where φ is an L-sentence and Ψ is a finite collection of L-sentences. *Soundness* would mean that there only exists valid proofs for statements that are true; *completeness* would mean that every true statement has a proof.

[4]That it preserves the trace also ensures that it is an isometry with regard to the operator norm, as the operator norm we use in L_{vNa}-structures is actually derived from a representation of the von Neumann algebra using the trace.

In continuous logic, entailment works the way you would expect from the analogy between the systems; $\Psi \vdash \varphi$ where Ψ and φ are L_{vNa}-sentences if for every $\mathcal{M}$ such that $\mathcal{M} \vDash \Psi$ (i.e. $\psi^{\mathcal{M}} = 0$ for all $\psi \in \Psi$), $\mathcal{M} \vDash \varphi$ (i.e. $\varphi^{\mathcal{M}} = 0$). Soundness is also straightforward; if there exists $\mathcal{M}$ with $\mathcal{M} \vdash \Psi$ such that $\varphi^{\mathcal{M}} \neq 0$, then we should not be able to prove $\Psi \vdash \varphi$. Completeness, on the other hand, is less straightfoward, and we will revisit in in Section 4.2.1.

3. Computational Complexity

In this section we will briefly develop the basic notions of Computational Complexity Theory, then proceed to define the classes we are concerned with in a way highly specialized to our current task. We also briefly discuss the way that Mathematical Quantum Physics is modeled through von Neumann algebras as it becomes relevant.

3.1. **Modeling Computing.**

We briefly discuss the basics of how we model and study computation, and introduce one very important definition.

3.1.1. *Turing Machines.* The Turing machine is probably the most canonical model of computing. We will not spend too much time on them, except to say that they a string as input and may in a finite number of steps return a string as output, or otherwise continue forever without halting. We may also permit the machine to emit an output multiple times without halting in some cases. Any definition of general computing you can think of will probably be equivalent.[5]

3.1.2. *Languages & Complexity.* A set of strings over some finite alphabet (or equivalently a subset of $\mathbb{N}$) $\mathbf{L}$ is known as a language. We arrange languages into classes depending on their *complexity*, which is a broad term that usually boils down to "how hard is it to prove whether or not a string is in this language?" Our first complexity class is as follows:

Definition 3.1.1 (R)**.** A language $\mathbf{L}$ is called recursive or decidable and is in R if there exists a Turing machine $\boldsymbol{M}$ that can *decide* $\mathbf{L}$, i.e. answer the question *"is $z \in \mathbf{L}$?"* correctly and in finite time for any string z.

The canonical example of a language not in R is **Halt**, which includes only those strings which code for Turing machines that halt (by which we mean halt when given an empty string as their input). The problem of finding a Turing machine that decides **Halt** is known as *the halting problem*, and it is known to be impossible. An intuitive reason for this is that its in some sense "cheating" – solving the halting problem would also solve nearly any math problem with a "yes/no" answer, as you could simply detect whether a program that searches for proofs of "yes" ever finds one. The actual formal reason is that it creates a paradox: if $\boldsymbol{H}$ solves the halting problem, then what happens when $\boldsymbol{A}$ is programmed to run $\boldsymbol{H}$ on $\boldsymbol{A}$'s own code and then do the opposite of what $\boldsymbol{H}$ says $\boldsymbol{A}$ will do?

To house **Halt**, we define another class:

Definition 3.1.2 (RE)**.** A language $\mathbf{L}$ is called recursively enumerable and is a member of RE if there is a Turing machine $\boldsymbol{M}$ that always answers the question *"is*

[5]In fact, this is known as the Church-Turing thesis.

$z \in \mathbf{L}$" correctly *when it halts*, but that is only bound to halt and return a result in the case that $z \in \mathbf{L}$.

Clearly **Halt** $\in$ RE as we can achieve the above by simply running the machine coded by z until it halts.

3.1.3. *Efficiency.* Computational complexity is not just interested in computability but also in how efficient the computation is. Rather than use absolute numbers, we usually model efficiency mathematically in terms of *scaling* – how fast the number of instructions scales with the length of the input. For a Turing machine $\boldsymbol{M}$, if there exists some polynomial $p(n)$ such that the number of instructions before halting when given input z is less than $p(|z|)$, we say that $\boldsymbol{M}$ runs in time polynomial in z, and we say this even if $\mathbf{M}$ is passed additional inputs other than z at the start. When z is clear, we simply say that $\boldsymbol{M}$ runs in polynomial time.

3.2. **Interactive Proofs and Games.** We will formalize our relevant complexity classes directly in terms of nonlocal games.

Definition 3.2.1 (nonlocal game, strategy, value)**.** A game with L players, n questions, and k answers consists of a pair $\mathfrak{G} = (\pi, D)$, where π is a probability distribution on $[k]^L = \underbrace{\{1, ..., k\} \times \cdots \{1, ..., k\}}_{L \text{ times}}$, and $D : ([k] \times [n])^L \to \{0, 1\}$.

Conceptually, this represents a protocol in which a party $\boldsymbol{V}$ we call the verifier randomly chooses one of k questions to ask each of the players (or, alternatively, provers) who we denote by $\mathfrak{P}_\ell$, $\ell \in [L]$. Say that $\boldsymbol{V}$ sends each $\mathfrak{P}_\ell$ the question $q_\ell \in [k]$. Then, each $\mathfrak{P}_\ell$ chooses a response $a_\ell \in [n]$. Based on these responses, the players collectively "win" if $\boldsymbol{V}$ decides their answers are satisfactory, which is determined by $D(q_1, a_1, ..., q_\ell, a_\ell)$. If the value is 1, they win; otherwise, they lose.

A (general) strategy for a nonlocal game is a conditional probability $p\Big((a_\ell)_{\ell \in [L]} \mid (q_\ell)_{\ell \in [L]}\Big)$ which gives a probability that for each $\ell \in [L]$, when $\mathfrak{P}_\ell$ is asked q_ℓ it responds with a_ℓ. In general strategies, players are allowed full communication, whence a multiplayer nonlocal game is not such a useful concept on its own.[6] It will only become useful when we restrict communication through stricter conditions on strategies.

Before that we do that, however, we will finish this definition with our model for the interaction of strategies with games. The expected value of strategy p on game $\mathfrak{G}$, is defined as

$$\mathrm{val}(\mathfrak{G}, p) := \sum_{(q_\ell)_\ell \in [n]^L} \pi\Big((q_\ell)_{\ell \in L}\Big) \sum_{(a_\ell)_\ell \in [k]^L} D(q_1, a_1, ..., q_L, a_L) p\Big((a_\ell)_{\ell \in [L]} \mid (q_\ell)_{\ell \in L}\Big)$$

and conceptually gives the probability that the players win $\mathfrak{G}$ when using strategy p.

To see how this relates to the concept of computability, consider the following imaginary situation: verifier $\boldsymbol{V}$ wants to efficiently determine whether any given string $z \in \mathbf{L}$. It has access to a very powerful, oracle-like entity known only as $\mathfrak{P}$.

[6] If L provers are allowed full communication, they are working together towards precisely the same goal and so behave as one prover.

While $\boldsymbol{V}$ is a lowly Turing machine, $\mathfrak{P}$ has no computational bounds at all, and will readily answer any questions $\boldsymbol{V}$ has. The problem is that $\mathfrak{P}$ is only interested in "winning" and will try to convince $\boldsymbol{V}$ that $z \in \mathbf{L}$ whether or not it actually is. The challenge for $\boldsymbol{V}$ is to use $\mathfrak{P}$'s miraculous capabilities to determine the truth, using its own limited computational ability to check its work. With this in mind, we connect nonlocal games to this computational challenge with the following definition:

Definition 3.2.2 (efficient mapping)**.** For any fixed finite alphabet and a fixed number of players, we that a mapping $z \mapsto \mathfrak{G}_z = (\pi_z, D_z)$ from strings to nonlocal games is efficient if the following two conditions hold:

1. There is a Turing machine $\boldsymbol{V}_1$ that efficiently implements $z \mapsto \pi_z$, in the sense that given the string z and a random string r, it can, in time polynomial in $|k|$, generate L questions $q_1, ..., q_L$ in such a way that the distribution of questions when fixing z is identical to that given by π_z.
2. There is Turing machine $\boldsymbol{V}_2$ that efficiently implements $z \mapsto D_z$, in the sense that, when given z, the questions $(q_\ell)_{\ell \in L}$, and the answers $(a_\ell)_{\ell \in L}$, it can, in time polynomial in $|z|$, output $D_z(q_1, a_1, ..., q_L, a_L)$.

The scenario described above is modeled (informally) by setting $L = 1$ and allowing the prover to use any strategy. We call it an *interactive proof system*, and the complexity class of all problems that can be probabilistically verified this way (or with an extended definition permitting multiple rounds of back-and-forth questions and answers) is denoted IP. A canonical example is the *graph isomorphism problem* with language **IsoGraph**. Given two graphs G_1, G_2 with an equal number of vertices and edges, it can be computationally intensive to determine whether the graphs are isomorphic (meaning that they are the same up to relabeling nodes) but there is an interactive proof verifiable in polynomial time. $\boldsymbol{V}$ randomly chooses one of the two graphs G_i, shuffles its node labels randomly to get G', and then shows it to $\mathfrak{P}$, asking whether G' is a shuffled copy of G_1 or of G_2. When G_1 and G_2 are not isomorphic, $\mathfrak{P}$ will be able to immediately determine which of the two was shuffled to get G', but when they *are* isomorphic $\mathfrak{P}$ will have to guess; thus, $\boldsymbol{V}$ can use $\mathfrak{P}$'s performance at this guessing game as a probabilistic proof of membership in **IsoGraph**.

3.2.1. *Multiple Interactive Provers.* The MIP-protocol is much like the scenario described above, except that it has two players who are not allowed to coordinate. In order to model this we introduce a new restriction on strategies: a 2 player, n-question, k-answer strategy p is called deterministic if there are functions $P_\ell : [n] \to [k]$, $\ell \in \{1, 2\}$ such that $p(P_1(q_1), P_2(q_2) \mid q_1, q_2) = 1$ for all $(q_1, q_2) \in [n]^2$. We denote the set of deterministic strategies with these parameters by $C_{\text{det}}(n, k) \subseteq [0, 1]^{n^2 k^2}$. We then say that the classical value of the game $\mathfrak{G}$ is

$$\text{val}(\mathfrak{G}) := \sup_{p \in C_{\text{det}}(n,k)} \text{val}(\mathfrak{G}, p).$$

We can then define MIP as follows:

Definition 3.2.3 (MIP)**.** A language $\mathbf{L}$ belongs to complexity class MIP if there is an efficient mapping $z \mapsto \mathfrak{G}_z$ such that:

- If $z \in \mathbf{L}$, then $\mathrm{val}(\mathfrak{G}_z) \geq \frac{2}{3}$.
- If $z \notin \mathbf{L}$, then $\mathrm{val}(\mathfrak{G}_z) \leq \frac{1}{3}$.

We fix $L = 2$ in the above definition as it turns out that analogous definitions but with more than two players will result in the same class. As such, from here on out we only consider 2-player games.

3.2.2. *Multiple Interactive Provers, Star.* We now introduce a new kind of strategy, and we start with a little bit of quantum physics. We can represent physical systems as Hilbert spaces, by $\mathcal{H}$. Any state the system can be in is an element $\xi \in \mathcal{H}$. To measure the system, say that the measurement has $m \in \mathbb{N}$ possible outcomes. To represent this measurement, we fix a collection of projections[7] $(P_i)_{i\in[m]}$ in $\mathcal{H}$ such that $\sum_{i=1}^{m} P_i = \mathrm{Id}_{\mathcal{H}}$, and say that the probability that outcome $i \in [m]$ occurs is equal to $\langle P_i\xi, \xi\rangle$. This is called a projection-valued measure (PVM), as we can can consider it as specifying a measure $\{i_1, ..., i_l\} \mapsto \sum_{j=1}^{l} P_i$.

To consider the composite system of $\mathcal{H}_A$ and $\mathcal{H}_B$, we turn to $\mathcal{H}_A \otimes \mathcal{H}_B$. For $\xi_A, \eta_A \in \mathcal{H}_A$, $\xi_B, \eta_B \in \mathcal{H}_B$, the presence of elements like $\xi_A \otimes \xi_B + \eta_A \otimes \eta_B$ in this space models *quantum entanglement*. We then can model two scientists, Alice and Bob, measuring physically separated but potentially entangled systems by fixing a state $\xi \in \mathcal{H}_A \otimes \mathcal{H}_B$, a PVM $(P_i)_{i\in[m_A]}$ on A, and a PVM $\left(Q_j\right)_{j\in[m_B]}$ on B, and saying that the probability of Alice receiving outcome $i \in [m_A]$ and Bob recieving outcome $j \in [m_B]$ after they perform their respective measurements is $\left\langle P_i \otimes Q_j\xi, \xi\right\rangle$.

We use these to develop quantum strategies, which represent a situation in which two players are not permitted to communicate but are granted entangled particles whose measurements they may use to coordinate.

Definition 3.2.4 (quantum strategy, entangled value, MIP*)**.** We will call a 2-player n-question, k-answer strategy p a quantum strategy if there are *finite dimensional* Hilbert spaces $\mathcal{H}_A$, $\mathcal{H}_B$, $\xi \in \mathcal{H}_A \otimes \mathcal{H}_B$, a collection of PVMs $\left(P_a^{(q)}\right)_{a\in[k]}$, $q \in [n]$ on $\mathcal{H}_A$, and a collection of PVMs $\left(Q_a^{(q)}\right)_{a\in[k]}$, $q \in [n]$ on $\mathcal{H}_B$ such that

$$p(a_1, a_2 \mid q_1, q_2) = \left\langle P_{a_1}^{(q_1)} \otimes Q_{q_2}^{(a_2)}\xi, \xi\right\rangle.$$

We can think about this as q_ℓ telling $\mathfrak{P}_\ell$ which measurement to perform, and then responding with answer a_ℓ if they get the a_ℓ-th outcome from that measurement.

The set of all such strategies will be denoted by $C_q(n, k) \subseteq [0, 1]^{n^2k^2}$. The entangled value of a game $\mathfrak{G}$ is then

$$\mathrm{val}^*(\mathfrak{G}) := \sup_{p\in C_q(n,k)} \mathrm{val}(\mathfrak{G}, p).$$

Finally, a language $\mathbf{L}$ belongs to the complexity class MIP* if there is an effective mapping $z \mapsto \mathfrak{G}_z$ such that:

- If $z \in \mathbf{L}$, $\mathrm{val}^*(\mathfrak{G}_z) \geq \frac{2}{3}$.
- If $z \notin \mathbf{L}$, $\mathrm{val}^*(\mathfrak{G}_z) \leq \frac{1}{3}$.

[7]We actually usually allow these to be positive operators, but restricting to projections is sufficient for our purposes.

3.2.3. *Other Quantum Strategies.* We will define a few other important varieties of strategies before we move on. Recall that in Definition 3.2.4, we have restricted outselves to finite-dimensional Hilbert spaces. As it happens, this restriction is in some ways not necessary. We will call a strategy as in the above definition but in which the Hilbert spaces are separable and infinite dimensional a quantum spacial strategy, and denote the set of such strategies by $C_{\text{qs}}(n,k) \subseteq [0,1]^{n^2k^2}$.

Clearly, we can consider $C_q(n,k) \subseteq C_{\text{qs}}(n,k)$, as any mere quantum strategy is also a quantum spacial strategy that happens to only use finitely many dimensions. Moreover, it can be shown that any quantum spacial strategy is a limit (using the topology inherited as a subset of $[0,1]^{n^2k^2}$) of quantum strategies using projections onto finite-dimensional subspaces, and so

$$C_{\text{qa}}(n,k) := \overline{C_{\text{qs}}(n,k)} = \overline{C_q(n,k)}.$$

As we are considering supremums, the distinction between these sets of strategies does not matter much, so they will mostly be used interchangably.

Another important distinction is that of being *synchronous* – a strategy p is synchronous if for every $q \in [n]$, there are $a_1^{(q)}, a_2^{(q)} \in [n]$ such that $p\left(a_1^{(q)}, a_2^{(q)} \mid q, q\right) = 1$. In other words, synchronous strategies are those in which asking the two players the same question determines (in the sense of being non-probabilistic) the player's answers. We will use $C_q^s(n,k)$ to denote synchronous quantum strategies, and $\text{sval}^*(\mathfrak{G}) = \sup_{p \in C_q^s(n,k)} \text{val}(\mathfrak{G}, p)$, and likewise do the same for C_{qa}^s and C_{qs}^s. Note that clearly we always have

$$\text{sval}^*(\mathfrak{G}) \leq \text{val}^*(\mathfrak{G}).$$

3.3. **MIP* = RE.** It is fairly trivial to show that MIP* $\subseteq$ RE; for a language $\mathbf{L} \in$ MIP*, we simply iterate through all possible strings, all possible finite-dimensional Hilbert spaces (which effectively just requires varying the dimension d), and then through a countable, dense subset of compatible quantum strategies. Simulating each of these strategies on each $\mathfrak{G}_z$, any time we have that $\text{val}(\mathfrak{G}_z, p) > \frac{1}{3}$, we determine that $z \in \mathbf{L}$, but if $\text{val}(\mathfrak{G}_z, p) \leq \frac{1}{3}$ we only know that this one strategy does not work and have not shown $z \notin \mathbf{L}$; hence, this shows MIP* $\subseteq$ RE and not MIP* $\subseteq$ R.

Then, in [7], Ji Zhengfeng et al. showed that MIP* = RE. To do this, it was sufficient that they find an efficient mapping $z \mapsto \mathfrak{G}_z$ that serves as a proof system for the language **Halt**, as the ability to determine whether any Turing machine halts can be used to determine whether or not an arbitrary Turing machine outputs a given string, thus determining membership for any RE-language. The paper indeed contains such a proof system, notwithstanding that their soundness parameter is slightly higher.

However, what they actually were able to prove was, in a few technically but very important ways, stronger than that:

Theorem 3.3.1 (MIP* = RE, [7])**.** *There exists an efficient mapping $z \mapsto \mathfrak{G}_z$ such that:*

- *If $z \in$ **Halt**, $\text{sval}^*(\mathfrak{G}_z) = 1$.*

- *If* $z \notin \mathbf{Halt}$, $sval^*(\mathfrak{G}_z) \leq \frac{1}{2}$.

4. From CEP To Computability

We now begin to bridge our previous sections together, following Goldbring and Hart's computability and model theoretic reformulation of the CEP as in [8].

4.1. **From CEP to Model Theory.** We can now restate the CEP, as given in Definition 1.3.5, in a way that is more amenable to solution via model theory:

Theorem 4.1.1 (Model-Theoretic Restatement of the CEP; [1] Section 7.2)**.** *The CEP is equivalent to the statement that* $Th_\forall(\mathcal{R})$ *is the unique universal theory of* II_1 *factors; that is, that for every* II_1 *factor* $\mathcal{M}$, $Th_\forall(\mathcal{M}) = Th_\forall(\mathcal{R})$.

Proof. The CEP states that for every II_1 factor $\mathcal{M}$, $\mathcal{M}$ embeds into $\mathcal{R}^{\mathfrak{U}}$ for some $\mathfrak{U}$. By Corollary 2.1.9 and the correspondance between classical and continuous Model Theory, this is precisely equivalent to $\mathrm{Th}_\forall(\mathcal{M}) \leq \mathrm{Th}_\forall(\mathcal{R})$. Then, as $\mathcal{R}$ embeds into every II_1 factor, for any such $\mathcal{M}$ we have that $\mathrm{Th}_\forall(\mathcal{R}) \leq \mathrm{Th}_\forall(\mathcal{M})$, and so we may equivalently restate this as $\mathrm{Th}_\forall(\mathcal{M}) = \mathrm{Th}_\forall(\mathcal{R})$. □

4.2. **Computability.** We now turn our attention to computability. When dealing with classical logic, we usually call an L-theory T computable if there is some algorithm that, given some L-sentence σ, can determine the answer to the question "is σ in T?" in finite time.

This, however, fails for continuous logic. The "truth values" of sentences in a continuous logic is some arbitrary real number–a real number that can ω-many bits to describe. To deal with this, we decide that we have obtained a real number through computation if, when asked, we can give approximations of arbitrary (but finite) precision. In real terms, this means:

Definition 4.2.1 (computable L_{vNa}-theory)**.** We say that a L_{vNa}-theory T is computable if there exists some algorithm that, given a L_{vNa}-sentence σ in the domain of T, can in finite time return ε-tight bounds for the valuation of σ in T – that is, rationals a, b such that $b - a < \varepsilon$ and $T(\sigma) \in (a, b)$.

We want to show that if the CEP is true, $\mathrm{Th}_\forall(\mathcal{R})$ is computable. By the above, this effectively means finding a method of computing arbitrarily good upper and lower bounds on $\mathrm{Th}_\forall(\mathcal{R})(\sigma) = \sigma^{\mathcal{R}}$, where σ is a universal L_{vNa}-sentence.

4.2.1. *Proof Systems and Completeness.* We now pick back up the discussion of a proof system for continuous logic given in Section 2.2.4. Again, we must move from exactitude of true or false to approximacy of real numbers. To talk about bounds, we introduce the notation

$$x \dot{-} y := \max(x - y, 0).$$

If we prove $\Phi \vdash \sigma \dot{-} r$, for an L_{vNa}-sentence σ, we have effectively found an upper bound on $\sigma^{\mathcal{M}}$ across all $\mathcal{M}$. We now give the completeness theorem we will be using, proven by [9] but in the restricted form relevant to our situation as given in [1]: for every L_{vNa}-sentence σ, we have

$$\sup\{\sigma^{\mathcal{M}} : \mathcal{M} \text{ a } \mathrm{II}_1 \text{ factor}\} = \inf\{r \in \mathbb{Q}^{>0} : T_{\mathrm{II}_1} \vdash \sigma \dot{-} r\}$$

where T_{II_1} is a recursively enumerable set of sentences such that $\mathcal{M}$ is a II_1 factor if and only if $\mathcal{M} \models T_{\mathrm{II}_1}$. In essence, this says that our upper bounds are tight; the infinum of the provable upper bounds is in fact equal to the greatest upper bound of the actual values.

With the knowledge that T_{II_1} is itself RE, we can work on our bounds.

4.2.2. *Upper Bound.* Finding the upper bound is pretty easy – all we really need to do is show that we can actually write $\sigma \dot{-} r$ as a universal sentence. First, note that that this function is continuous. Moreover, recall that a universal L_{vNa}-sentence σ is a sentence of the form

$$\left(\sup_{x_{i_1}}\right)\cdots\left(\sup_{x_{i_n}}\right) f(\varphi_1(\bar{x}),\cdots,\varphi_m(\bar{x}))$$

where f is a continuous function, $\varphi_j(\bar{x})$ is a basic L_{vNa}-formula for $j \in \{1,...,m\}$, and $f(\varphi_1(\bar{x}),...,\varphi_m(\bar{x}))$ has positive range. Thus, writing

$$\sigma \dot{-} r := \left(\sup_{x_{i_1}}\right) f(\varphi_1(\bar{x}),\cdots,\varphi_m(\bar{x})) \dot{-} r$$

we have thus written $\sigma \dot{-} r$ as a universal L_{vNa}-sentence, checking against Definition 2.2.1.

We can thus proceed as follows:

Theorem 4.2.2. *For any universal L_{vNa}-sentence σ, we can compute successively tighter upper bounds for $\sigma^{\mathcal{R}}$ such that, for any $\varepsilon > 0$, our guess eventually (i.e. in finite time) is ε-close to the actual value.*

Proof. Find some algorithm that enumerates all L_{vNa}-proofs in T_{II_1}. Each time we find a proof of the form $T_{\mathrm{II}_1} \vdash \sigma \dot{-} r$ where r is a positive rational number, we learn that $\sigma^{\mathcal{M}} \leq r$ for all II_1 factors $\mathcal{M}$, by soundness. In particular, we learn that $\sigma^{\mathcal{R}} \leq r$ Each time we prove this for a lower value of r, we note it down, as we have found a tighter upper bound for $\sigma^{\mathcal{R}}$.

As $\mathcal{R}$ embeds into any II_1 factor and σ is universal, we have that $\sigma^{\mathcal{M}} \leq \sigma^{\mathcal{R}}$. As $\mathcal{R}$ is a II_1 factor, $\sup\{\sigma^{\mathcal{M}} : \mathcal{M} \text{ a } \mathrm{II}_1 \text{ factor}\} = \sigma^{\mathcal{R}}$. Moreover, by completeness of our proof system, $\sup\{\sigma^{\mathcal{M}} : \mathcal{M} \text{ a } \mathrm{II}_1 \text{ factor}\} = \inf\{r : T_{\mathrm{II}_1} \vdash \sigma \dot{-} r\}$. We thus can guarantee that sooner or later, this process will find an ε-close approximation. □

4.2.3. *Lower Bound.* Finding a lower bound is a little harder. The process in Theorem 4.2.2 will not work for lower bounds without modification: actually negating σ turns its supremums into infimums, and so, as we have defined it, $(r \dot{-} \sigma)^{\mathcal{M}} \neq \max(r - \sigma^{\mathcal{M}}, 0)$. We also cannot use $\min(\sigma - r, 0)$, as for a sentence to be universal, its range needs to be nonnegative. Instead, we use the following trick:

Recall the definition of a basic L_{vNa}-formula given in Definition 2.2.1. There are only three atomic functions used to obtain real numbers from elements of $\mathcal{M}_1$, $\mathrm{tr}_{\mathfrak{R}}$, $\mathrm{tr}_{\mathfrak{I}}$, and d. Recalling that we define trace to be normalized with $\mathrm{tr}(\mathrm{Id}_{\mathcal{H}}) = 1$, the maximum value that either the real or imaginary trace may output is 1, and as we

restrict ourselves to the unit ball $\mathcal{M}_1$ of each $\mathcal{M}$, the maximum $d(a,b)$ may output on any $a,b \in \mathcal{M}_1$ is also 1. As such, for any sentence σ, without even considering von Neumann algebras, we may calculate an absolute maximum M_σ on $\sigma^{\mathcal{M}}$ across all $\mathcal{M}$ by replacing every occurrence of $\mathrm{tr}_{\mathfrak{R}}(t)$. $\mathrm{tr}_{\mathfrak{J}}(t)$, and $d(t,t')$ (where t and t' are an L_{vNa}-terms) with 1, and then calculating it out.

With that, we can calculate our lower bound:

Theorem 4.2.3. *For any universal L_{vNa}-sentence σ, we can compute successively tighter lower bounds for $\sigma^{\mathcal{R}}$ such that, for any $\varepsilon > 0$, our guess eventually (i.e. in finite time) is ε-close to the actual value.*

Proof. We perform the same procedure as in the proof for Theorem 4.2.2, but instead we find greater and greater values for r such that $T_{\mathrm{II}_1} \vdash M_\sigma \dot{-} \sigma \dot{-} r$. Each time we find a value of r that satisfies this, we know that $\sigma^{\mathcal{R}} \geq M_\sigma - r$, and again, by completeness, we eventually get ε-close. □

Putting this together, we show the following:

Theorem 4.2.4 ([1], Theorem 7.1)**.** *If we assume the CEP, this implies that the universal theory of the hyperfinite II_1 factor is computable.*

Proof. We run the algorithms we use in Theorem 4.2.3 and Theorem 4.2.2 simulatenously, until we obtain results that are ε-close, for the given ε. □

5. From MIP* To Non-Computability

Our last task is to demonstrate that $\mathrm{Th}_\forall(\mathcal{R})$ cannot possibly be computable:

Theorem 5.0.1 ([10], as cited from Theorem 7.2 in [1])**.** *The universal theory of $\mathcal{R}$ is not computable.*

To do this, we use the oldest trick in the book: we show that if it were, we could decide **Halt**.

We know from Theorem 3.3.1 that there is an efficient mapping of Turing machine codes z to games $\mathfrak{G}_z$ such that $\mathrm{sval}^*(\mathfrak{G}_z) = 1$ when $z \in \mathbf{Halt}$ and $\mathrm{sval}^*(\mathfrak{G}_z) \leq \frac{1}{2}$ otherwise. Suppose that $\mathrm{sval}^*(\mathfrak{G}_z)$ could be represented as an L_{vNa} -sentence σ_z such that $\sigma_z^{\mathcal{R}^{\mathfrak{U}}} = \mathrm{sval}^*(\mathfrak{G}_z)$. Then by Theorem 4.2.4, we can compute an ε-tight bound on the value of $\sigma_z^{\mathcal{R}^{\mathfrak{U}}}$ in finite time. If we set $\varepsilon < \frac{1}{2}$, then this bound can only include one of $\frac{1}{2}, 1$; if the value is 1, we know $z \in \mathbf{Halt}$, and otherwise it is not. Thus, finding such a representation is sufficient to prove the CEP absurd.

Recall that

$$\mathrm{sval}^*(\mathfrak{G}) = \sup_{p \in C^s_{\mathrm{qa}}(n,k)} \sum_{q_1,q_2 \in [n]} \pi(q_1,q_2) \sum_{a_1,a_2 \in [k]} D(q_1,a_1,q_2,a_2) p(a_1,a_2|\ q_1,q_2).$$

This formula is in fact a linear function s of the values $p(a_1,a_2 \mid q_1,q_2)$, is clearly continuous and thus valid in a L_{vNa}-sentence. Thus, our next step is to find a representation of $p(a_1,a_2 \mid q_1,q_2)$ using $*$-polynomials in $\mathcal{R}_1^{\mathfrak{U}}$.

Luckily, the restriction to synchronous strategies makes this entirely possible. In [1], Goldbring cites a result from [11] (specifically Theorem 3.6) that can be used

to show that $p \in C^s_{\text{qa}}(n,k)$ if and only if there is a PVM $\left(f^a_q\right)_{q\in[k]}$ for each $a \in [n]$ such that $p(a_1,a_2 \mid q_1,q_2) = \operatorname{tr}_{\mathcal{R}^{\mathfrak{U}}}\left(f^{a_1}_{q_1} f^{a_2}_{q_2}\right)$. For a tuple $\bar{f} \in \mathcal{M}_1$ to be a PVM, we just need to check that each f_i is a projection, so $d(f,f^*) = d(f,f^2) = 0$, and that the tuple sums to $\mathrm{Id}_{\mathcal{H}}$, so $d\left(\sum_{i=1}^k f_i, 1\right) = 0$. We thus can easily define a L_{vNa}-formula

$$\psi_k(\bar{x}) \coloneqq \max\left(\max_{i\in[k]} d(x_i,x_i^*), \max_{i\in[k]} d(x_i,x_i^2), d\left(\sum_{i\in[k]} x_i, 1\right)\right)$$

such that $\psi_k(\bar{x})$ is satisfied (in the sense of being equal to zero) by all and only PVMs. Let $X_k \coloneqq \left\{\bar{f} : \psi_k\left(\bar{f}\right) = 0\right\}$.

We now have rewritten $\text{sval}^*(\mathfrak{G})$ as

$$\sup_{\left(\bar{f}^{(a)}\right)_{a\in[n]}\in X_k} s\left(\left(\operatorname{tr}_{\mathcal{R}^{\mathfrak{U}}}\left(f^{a_1}_{q_1} f^{a_2}_{q_2}\right)\right)_{a_m\in[k],q_m\in[n]}\right).$$

We are not yet done though, because quantification over specific subsets of the unit ball is not a syntactical element of L_{vNa}-formulas. In classical logic, this would be a technicality; X_k is described by formula $\psi_k(\bar{x})$, so we could just use

$$(\forall \bar{x})\left(\psi_k(\bar{x}) \to s\left(\left(\operatorname{tr}_{\mathcal{R}^{\mathfrak{U}}}\left(x^{a_1}_{q_1} x^{a_2}_{q_2}\right)\right)_{a_m\in[k],q_m\in[n]}\right)\right)$$

recalling that $\forall$ is the equivalent of sup. Unfortunately, this does not generally work in continuous logic—X_k being defined by ψ_k is not enough to ensure that the supremum over X_k of a L_{vNa}-formula can be written as a legitimate L_{vNa}-sentence. Fortunately, there are stronger criteria that *can* ensure that, and X_k *does* satisfy those!

The relevant notion is that of being a definable set. [1] gives that to meet this criteria we need to show that for all $\varepsilon > 0$ there exists a $\delta > 0$ such that for all $\bar{a} \in \mathcal{R}_1^{\mathfrak{U}}$, when $\psi_k(\bar{a})^{\mathcal{R}^{\mathfrak{U}}} < \delta$, there exists $\bar{b} \in \mathcal{R}_1^{\mathfrak{U}}$ with $\psi_k(\bar{b})^{\mathcal{R}^{\mathfrak{U}}} = 0$ and $d(\bar{a},\bar{b}) < \varepsilon$. That ψ_k satisfies these criteria is in fact Lemma 3.5 of [11].

Thankfully, the process of writing our supremum as an L_{vNa}-sentence is computable, and the resulting σ_k remains universal. We thus use CEP to solve the halting problem, and the CEP vanishes in a puff of logic.

References

[1] Goldbring, I. (2022, June 17). The Connes embedding problem: A guided tour. *Bulletin of the american mathematical society*, 59(4), 503–560. https://doi.org/10.1090/bull/1768

[8] Goldbring, I., & Hart, B. (2013, August 12). *A computability-theoretic reformulation of the Connes Embedding Problem*. (Retrieved May 14, 2023, from http://arxiv.org/abs/1308.2638)

[10] Goldbring, I., & Hart, B. (2021, June 21). *The Universal Theory Of The Hyperfinite II$_1$ Factor Is Not Computable*. https://doi.org/10.48550/arXiv.2006.05629

[3] Hodges, W. (1997). *A shorter model theory*. Cambridge University Press.

[7] Ji, Z., Natarajan, A., Vidick, T., Wright, J., & Yuen, H. (2022, November 4). *MIP*=RE*. (Retrieved May 14, 2023, from http://arxiv.org/abs/2001.04383)

[2] Jones, V. F. R. (2015, November 13). *Von Neumann Algebras*. https://cdn.vanderbilt.edu/vu-my/wp-content/uploads/sites/3392/2020/10/15162810/vonNeumann2015.pdf

[5] Keisler, H. J. (2010). The ultraproduct construction. In *Contemporary Mathematics* (Vol. 530). American Mathematical Society. https://doi.org/10.1090/conm/530/10444

[11] Kim, S.-J., Paulsen, V., & Schafhauser, C. (2018, March 16). A synchronous game for binary constraint systems. *Journal of mathematical physics*, 59(3), 32201. https://doi.org/10.1063/1.4996867

[4] Kunen, K. (1972). Ultrafilters and Independent Sets. *Transactions of the american mathematical society*, 172, 299–306. https://doi.org/10.2307/1996350

[6] Tserunyan, A. (n. d.). *Saturation of Ultraproducts*. https://www.math.mcgill.ca/atserunyan/Courses/2014F.Math570.Logic/saturation.pdf

[9] Yaacov, I. B., & Pedersen, A. P. (2010, March). A proof of completeness for continuous first-order logic. *The journal of symbolic logic*, 75(1), 168–190. https://doi.org/10.2178/jsl/1264433914

McGill University, Montreal, QC
Email address: joel.newman@mail.mcgill.ca
URL: alternating.group