SEPARATING FORMAL SYSTEMS: RCA_0, WKL_0, AND ACA_0

JUDSON BUCHKO

Abstract. This paper serves as an introduction to reverse mathematics and proof theory. To do this, we introduce the systems of RCA_0, WKL_0, and ACA_0, which correspond to different allowed systems of induction, comprehension, and algorithmic strength. Introducing these systems allow one to see what axioms certain mathematical theorems and properties are dependent on and equivalent to, and providing a good separation of them therefore provides a separation of the strength of our mathematics. We will discuss the definitions of these systems, why they are defined the way that they are, how they are separated, and some of the mathematical properties that are provable in each system.

Contents

1. RCA_0 and its Motivation

We will begin our discussion of reverse mathematics with defining the system that all other systems will be based on, RCA_0, as well as providing the motivation for its axioms in the form of a discussion of computability theory. Some definitions will be required before we define RCA_0 itself.

We begin by defining the two-sorted language of *second order arithmetic*.

Definition 1.1. The language of second order arithmetic, commonly denoted as Z_2, is a mathematical language consisting of two kinds of variables, 'individuals' which represent numbers and are often denoted as lower case letters, i.e. a, and 'set variables', which may be thought of as sets of natural numbers and are denoted by capital letters, i.e. A. The language contains a symbol for set membership, $\in$, as well as the existential and universal quantifiers $\exists$ and $\forall$ for quantification of individuals and set variables. Individual terms are defined in the language and the structure of second order arithmetic by the constant 0 and successive applications of the successor function S, which adds 1 to the input. Additionally, the language

Date: DEADLINES: Draft AUGUST 14 and Final version AUGUST 28, 2024.

includes the nonlogical symbols $+, *, <$ which specify the operations and relations they classically do, and finally an equality symbol $=$. Thus, in notation, the language of second order arithmetic is given the signature $\mathcal{L} = (0, S, +, *, =, <, \in)$

For example, a formula in the language of second order arithmetic is: $\forall A(\exists x(x \in A) \vee A = \emptyset)$.

The systems of RCA_0, WKL_0, and ACA_0 are, more precisely, subsystems of second order arithmetic, where a subsystem of second order arithmetic is a collection of theorems in $\mathcal{L}$ that are stated as axioms of the *theory* of the subsystem. What is critical in the discussion of these systems of *second order arithmetic* are *formulas.* These are the formalization of a common kind of mathematical sentence where we are allowed to have implications, quantifiers, parameters, and things of that like. An *arithmetic formula* is a formula that does not quantify over sets, for instance, $t < u$, $t = u$, and so on. The language of second order arithmetic allows us access to *second order objects*, these are sets. We must be careful, however, as second order arithmetic contains no term for forming operations for sets, only the terms for set variables themselves. Because of this, we work with sets by doing operations and making reference to their first order elements. For finite sets, we can do a little better and 'code' the entire set into one first order element in such a way that the information of all the elements in the set can be recovered through operations on the 'code' of the finite set. As will be seen, there is an effective coding that can be done in RCA_0. between finite sets and numbers to reduce second-order problems to, often simpler, first order ones that will be discussed in detail in section 1.3.

Our formulae are often abbreviated as just ϕ, or ψ, where $\phi(x)$ is a certain instance of the formula holding for the free variable x.

Sets are however fine to be used as parameters for our formulas, and where pertinent, a set parameter will be indicated alongside free variables and other first-order parameters in the following way. For a set X, parameters $y_1, y_2, ..., y_n := y$, and free variables $x_1, x_2, ..., x_n := x$, we write $\phi(x, y, X)$ for the instance of the formula with those parameters and free variables.

The classification of formulas is done along the basis of the quantifiers employed in the formula. The most basic kind of quantifier is the *bounded quantifier*, of the form $\forall x < t$, or $\exists x < t$. The most basic type of formula that we consider is the *bounded quantifier formula* or the Σ^0_0 formula, a formula quantified only over bounded quantifiers, for instance, $(\forall x < t)[t - x > 0]$. We distinguish other kinds of formulas by their number and order of *unbounded quantifiers*, this kind of distinguishing is known as the *Arithmetic Hierarchy.*

Definition 1.2. A Σ^0_n formula is of the form $\exists x_1 \forall x_2 \exists x_3 \forall x_4 ... Q x_n \phi$ where ϕ is Σ^0_0, and Q is $\exists$ if n is odd, and is $\forall$ if n is even. A Π^0_n formula is one of the form $\forall x_1 \exists x_2 \forall x_3 \exists x_4 ... Q x_n \phi$ where ϕ is Σ^0_0, and Q is $\forall$ if n is odd, and is $\exists$ if n is even.

We now have enough definitions taken care of to start defining RCA_0. RCA_0 comes with the base axioms of Peano Arithmetic, or P_0, as well as an induction scheme and a comprehension scheme that tells us what kind of sets we can talk about and construct in models of RCA_0

Definition 1.3. P_0^- refers to just the first order axioms of Peano arithmetic over a discrete, ordered, commutative semiring. It consists of the closures of the following axioms over such a structure:

$x+y=y+x \quad x\cdot y=y\cdot x$	(Commutativity)
$(x+y)+z=x+(y+z) \quad (x\cdot y)\cdot z=x\cdot(y\cdot z)$	(Associativity)
$x(y+z)=x\cdot y+x\cdot z$	(Distributivity)
$x+0=x \quad x\cdot 0=0 \quad x\cdot 1=x$	(Identity and Absorption)
$\forall x(\neg(x<x))$	($<$ is Irreflexive)
$x<y\wedge y<z\rightarrow x<z$	($<$ is Transitive)
$x<y\vee y<x\vee x=y$	($<$ is Trichotomous)
$(x<y)\rightarrow(x+z<y+x)$	(Ordering Preserved Under Addition)
$(0<z\wedge x<y)\rightarrow(x\cdot z<y\cdot z)$	(Ordering Preserved Under Positive Multiplication)
$(x<y)\rightarrow\exists z(x+z=y)$	(The Larger Equals the Smaller plus Another.)
$0<1$	(0 and 1 are Distinct.)
$0\leq x$	(0 is the Minimum Element.)
$(0<x)\rightarrow(1\leq x)$	(0 is covered by 1)

Over this basic structure of axioms, RCA_0 allows for two types of induction, first, the one packaged together with Peano Arithmetic.

Definition 1.4. P_0, or Peano arithmetic with restricted induction, is P_0^- with the addition of *set induction*, which states:

$$(0\in X\wedge\forall n[n\in X\rightarrow n+1\in X])\rightarrow\forall n(n\in X)$$

Second, a restricted type of induction over formulae, limiting us to only Σ_1^0 formulas for which we can make inductive conclusions over.

Definition 1.5. The Σ_1^0 induction scheme states that for each Σ_1^0 formula ϕ, with distinguished free variable n:

$$(\phi(0)\wedge\forall n[\phi(n)\rightarrow\phi(n+1)])\rightarrow\forall n(\phi(n))$$

Note that Σ_1^0 induction, and in fact Π_1^0, Σ_n^0, and Π_n^0 induction are downward closed for the kinds of formulas that can be inducted on. For instance, if I have Σ_1^0 induction and I want to induct on a Σ_0^0 or equivalently Π_0^0 formula ψ, I can just add a dummy quantifier to my formula. For instance, $\exists x\ \psi$ where x is neither a parameter or free variable in ψ, now $\exists x\ \psi$ is Σ_1^0 and can be inducted as such. The same reasoning holds as for why Σ_n^0 induction or similarly higher inductive schemes imply all lower forms of induction: just add dummy quantifiers to your lower formulae.

RCA_0 also restricts the kinds of sets that can be constructed within it. It does this through a restricted *comprehension scheme*, where a comprehension scheme is able to take the elements that satisfy a given formula and *comprehend* them into a set. There are at least as many comprehension schemes as there are classifications of formula, but the one we highlight is Δ_1^0 comprehension.

Definition 1.6. The Δ_1^0 comprehension scheme consists of, for pairs of formulas ϕ, where ϕ is Σ_1^0, and ψ, where ψ is Π_1^0:

$$\forall n[\phi(n)\leftrightarrow\psi(n)]\rightarrow\exists X\forall n[n\in X\leftrightarrow\phi(n)]$$

Putting it all together, we come to the definition of RCA_0, the weakest base system for reverse mathematics.

Definition 1.7. The system RCA_0 consists of P_0 with Σ_1^0 induction and Δ_1^0 comprehension.

The choices for what kinds of induction is allowed and what kinds of sets are allowed to be constructed may seem arbitrary at first. For instance, why Σ_1^0 induction and not Σ_5^0? Or, why not Π_1^0 comprehension instead? The answers to these questions come in the name of RCA_0 itself, where RCA stands for the Recursive Comprehension Axiom, which is exactly the same as Δ_1^0 comprehension, and in modern terms may be more accurately called the "Computable Comprehension Axiom". For the motivation behind RCA_0 lies in computability theory.

1.1. **Computability Theoretic Motivation.**

This paper will assume basic computability-theory knowledge of the concepts of the Turing machine, the oracle Turing machine, etc. For this section, we will not explicitly prove that our discussion of computability theory can be done in RCA_0, instead, this is more of an informal section that motivates RCA_0's definition, so we will talk in computability-theoretic terms rather than reverse mathematical ones until we start making our connections.

Definition 1.8. A Countably Enumerable, or c.e., set A is the domain of a computable function Φ

We will now provide multiple equivalent characterizations of a c.e. set, the first should sound familiar.

Definition 1.9. A set is in Σ_1^0 form if it is the projection of a computable relation R, where a projection A takes the following form $A = \{y : (\exists x)R(x, y)\}$

Here we have the familiar 'one unbounded existence quantifier' form of the Σ_1^0 formula. However, instead of being followed by a Σ_0^0 bounded quantifier predicate, it is followed by a computable predicate, which may seem like an incongruence at first, but is really hinting at something deeper. The general computable function f or relation R may be seen, by definition, as a Turing machine that produces the same outputs as f on the same inputs, like, $(\exists s)\phi_{e,s}(x) = f(x)$, where s is the number of steps the Turing machine will run, halting or not. If there exists an s such that the above is true for a given x, and one exists for every x, then we say f is a computable function. We can put the existential quantifier in the function/relation together with the existential quantifier with $\exists x$ together in a block like so: $A = \{y : (\exists\langle x, s\rangle)\phi_{e,s}(x)$ such that we have one coded quantifier followed by a Turing machine that will run for only finitely many steps, something with only bounded quantifiers. Thus, it seems accurate to refer to such a set as in Σ_1^0 form.

Theorem 1.10. *A set is c.e. iff it is* Σ_1^0

Proof. First, if A is c.e. then $A = \text{dom}(\Phi_e)$ where Φ is a Turing machine with index e. Thus,

$$x \in \text{dom}(\Phi_e) \leftrightarrow (\exists s)[x \in \text{dom}(\Phi_{e,s})]$$

where the s is the number of steps that the turing machine will run, determined before the turing machine starts, where we halt no matter what once we've reached s steps in the operation of the machine. The relation $\{\langle x, e, s\rangle : x \in \mathrm{dom}(\Phi_{e,s})\}$ where the angle brackets represent a computable coding to ω, where ω is the naturals, s.t. x, e, s can be recovered from their code in ω. This relation itself is in fact computable, since we just calculate on x until the program halts, or until we have computed for s steps. Thus, since we can recover x in a computable fashion from this computable relation, A is a projection of a computable relation R, so A is Σ^0_1.

In the opposite direction, if A is Σ^0_1 then $A = \{y : (\exists x)R(x, y)\}$ where R is computable, then $A = \mathrm{dom}(\psi)$ where $\psi(x) =$ The least y such that $R(x, y)$ ψ is computable because R is, so A is *c.e.* □

We will now look at another equivalent characterization of *c.e.* that will, together with the first characterization, will provide sufficient motivation for Σ^0_1 induction.

Theorem 1.11. *For a set A. A is c.e. iff A is the range of some computable function Ψ, or $A = \emptyset$.*

Proof. First, the backwards direction. If $A = \emptyset$ then A is vacuously c.e. Otherwise, suppose $A = \mathrm{range}(\Psi_e)$. The set $\mathrm{range}(\Psi_e)$ is equivalent to the set $\{y : (\exists s)(\exists x)[\psi(x)_{e,s} = y]$ This is a computable relation, and the two quantifiers can be compressed into one using a computable coding, so A is of Σ^0_1 form, so A is c.e.

Now, the forwards direction, let $A = \mathrm{dom}(\phi_e) \neq \emptyset$. Choose an $a \in \mathrm{dom}(\phi_e)$. Define the computable function f by:

$$f(\langle s, x\rangle) = \begin{cases} x & \text{if } x \in \mathrm{dom}(\phi_{e,s+1}) - \mathrm{dom}(\phi_{e,s}) \\ a & \text{if } x \notin \mathrm{dom}(\phi_{e,s+1}) - \mathrm{dom}(\phi_{e,s}) \end{cases}$$

f's computability is clear, since calculating the domain of a finite-step Turing machine is a computable process. Each $x \in \mathrm{dom}(\phi_e)$, $x \neq a$ is listed by this function exactly once, since f outputs x at the exact step at which ϕ_e goes from not halting to halting on x, which is only at one step s_x per x. So clearly, $\mathrm{range}(f) = A$. This completes the proof. □

With this in context, we can provide a motivation for Σ^0_1 induction. As, since the set of the x that satisfy a Σ^0_1 formula ϕ is a Σ^0_1 set, so is *c.e.*, and we can provide a computable listing, or enumeration, of that set with the computable function f in the second half of the above proof. So while we can't construct the set explicitly in RCA_0, we can count its elements, so the inductive conclusion that $(\forall x)\phi(x)$ is decided by a finitisitc, computable algorithm for each x.
Next, we'll provide a motivation for Δ^0_1 comprehension.

Definition 1.12. A set A is in Δ_1 form if A and A^c (A-Complement) are both Σ^0_1

Equivalently, since we can characterize A^c like $A^c = \{y : (\exists x)R(x, y)\}$, we can characterize A like $A = \{y : (\forall x)\neg R(x, y)\}$. Thus, we can say that a Δ_1 set is both Σ^0_1 and Π^0_1.

Theorem 1.13. *A set A is computable iff A and A^c are c.e., equivalently, A is Δ_1.*

Proof. If A is computable, then we know that A^c is computable, so A and A^c are both c.e.
If A is Δ_1, then A and A^c are c.e., so let, for some $e \in \omega$, $A = \mathrm{dom}(\phi_e)$ where

ϕ_e is a computable function. Similarly, $A^c = \text{dom}(\phi_i)$. We define the computable function

$$f(x) = \text{The least such s such that } \{x \in \text{dom}(\phi_{e,s}) \text{ or } x \in \text{dom}(\phi_{i,s})\}$$

Then we have that $x \in A$ iff $x \in \text{dom}(\phi_{e,f(x)})$ Now, we can set up the characteristic function $\chi_A(x)$ to output 1 only when $\phi_{e,f(x)}(x)\downarrow$, and 0 when $\phi_{i,f(x)}(x)\downarrow$ The reason this works is that every $x \in \omega$ will cause f to halt, so χ_A is a finite algorithm over all $x \in \omega$. This is unlike solely c.e. sets' characteristic function, where the members outside of the set can result in indeterminate calculation.

□

This proof provides sufficient motivation for Δ^0_1 comprehension, for now, we see that we are only able to make sets in RCA_0 if those sets are computable, i.e. have equivalent Σ^0_1 and Π^0_1 characterizations.

1.2. Models of RCA_0.

RCA_0, as it stands thus far, is just a system of schemes and allowed operations, it doesn't specify the first order system which it works over. Models specify the first order and second order parts over which a subsystem of second order arithmetic can reference, therefore we will differentiate between our systems by differentiating between models that model said systems.

Definition 1.14. A model is structure $\mathcal{M}$ where $\mathcal{M} = \{M, S, +_M, \cdot_M, 0_M, 1_M, \leq_M\}$ Where M is a set equipped with interpretations fo the operations $+$, $*$, etc, $S \subset \mathcal{P}(M)$ is the domain for quantifiers over set variables, basically the 'allowed' sets in the model; and the operations, elements, and relations are the allowed operations and significant field-elements in the model $\mathcal{M}$, lastly the relation $\leq_M$ implies an ordering.

However, not all models are created equal. For most of this paper, we will consider a model of RCA_0 that has ω as its first-order part and where S are only the sets which are constructible in RCA_0. This requires some prior definitions, the set join operation and the *Turing ideal*.

Definition 1.15. An ω-model is a model structure $\mathcal{M}$ in which $M = \omega$, and the arithmetical operations are the standard ones described in Definition 1.4.

The options for S given $M = \omega$ are exactly the Turing Ideals in the context of RCA_0.

Definition 1.16. The set join operation $\oplus$, when operating on sets A and B and forming the set $A \oplus B$ refers to the set:

$$A \oplus B = \{2n | n \in A\} \cup \{2n+1 | n \in B\}$$

Note that the set join operation is RCA_0 definable, that is the sets described by the set join operation are definable with the allowed methods. Apply Σ^0_0 comprehension to the formula $(\exists m < n)[(m \in A \land m+m = n) \lor (m \in B \land m+m+1 \in B)]$.

Definition 1.17. A Turing ideal is a nonempty set $I \subseteq \mathcal{P}(\omega)$ such that the following hold for all X, Y :

(1) If $X, Y \in I$, then $X \oplus Y \in I$
(2) If $X \in I$ and $Y \leq_T X$ then $Y \in I$

A Turing ideal is an 'ideal' over the Turing degrees, which has a more general structure we will not specify, but the idea is that Turing ideals are the exact type of structure that models the set-construction restrictions present in RCA_0 from a computability theory perspective, and allow us to define omega models of RCA_0, which are the most natural models to work in.

Theorem 1.18. *A structure* $\{\omega, S, +_M, \cdot_M, 0_M, 1_M, \leq_M\}$ *is a model of* RCA_0 *iff* $S \subset P(\omega)$ *is a Turing ideal.*

Proof. Consider a set X which is Δ^0_1 definable with parameters $Y_1, Y_2, ..., Y_m$, because the information contained within the parameter sets is essential to defining X in a computable fashion, a general Δ^0_1 set X is computable in the join of its parameters. This corresponds to closure property (2) of the Turing ideal, as if X is computable in the join of its computable parameters $\oplus_i Y_i \in I$, then $X \leq_T \oplus_i Y_i$ implies $X \in I$, or X is Δ^0_1 definable with those parameters. In the case of closure property (1), for two Δ^0_1 sets X, Z, the computability of their join is implied by the computability of the join of their parameters, which we know is true since the set join operation with finite parameters is Σ^0_0. □

Finally we take a brief detour to explore some of the implications of the existence of nonstandard models, and therefore see why we stick to an ω-model as the choice model.

Definition 1.19. The standard model of second order arithmetic is the model $\mathcal{M}$ such that $\mathcal{M}$ is an ω-model and $S = \mathcal{P}(\omega)$.

Models with non-standard first-order parts end up introducing a distinction between bounded and finite sets which must be parsed out with a coding scheme.

Definition 1.20. A nonstandard first order element is an element that lies outside of the natural numbers ω, call it c, that is "after" all the others, as it can't lie within the naturals as by PA it must then be the successor of one of them and be succeeded by another. So it must lie after all of them, $(\forall n \in \omega) c > n$. We will not go into details on the construction of nonstandard models, just state they exist.

Sets in RCA_0 are often coded with single elements, and because the model decides which first order elements are available to us, we speak of sets X as coded or bounded in relation to the model $\mathcal{M}$.
As we begin to prove things in RCA_0, we will need to formally reference our coding scheme, the choice of which will be critical for later theorems. We use the following definitions.

Definition 1.21. The bijective pairing function, $p : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is defined as $p(x, y) := \frac{1}{2}(x + y)(x + y + 1) + y)$. The notation (x, y) or $\langle x, y\rangle$ notates the output of the pairing function with inputs x, y.

The pairing function is the base tool for coding sets with numbers in RCA_0, and it is a primitive recursive (so computable) function, and so its output can be decided in RCA_0, as well as its inversion which allows one to recover the pair, forming an effective code for pairs.

Definition 1.22. A number (first order object) c represents a set X if there are $k, m,$ and n such that $c = (k, (m, n))$, and, for all i, we have $i \in X$ iff

$$(i < k) \wedge (m \cdot (i + 1) + 1 \text{ divides } n).$$

The code for a set X is the least such c that represents X in the above fashion. If such a c exists, X is coded. The above coding scheme is useful for working in RCA_0 since the membership of a given i in X can be decided by a formula with solely bounded quantifiers according to our definition, so a Σ^0_0 formula.

As previously stated, we view finite and bounded sets in relation to the model $\mathcal{M}$

Definition 1.23. Let $\mathcal{M}$ be a model of RCA_0, and $X \subset M$, (X is not necessarily in S).

(1) X is $\mathcal{M}$-bounded (or just "bounded", if the model is assumed), if there is a $n \in M$ such that $i \leq n$ for all $i \in X$.
(2) X is $\mathcal{M}$-coded (or just "coded") if there is a $c \in M$ that codes X as in Definition 1.22.

It remains to be shown that this coding scheme is effective in coding bounded sets in RCA_0

Theorem 1.24. *RCA_0 proves that for any set X, if X is $\mathcal{M}$-bounded, then X is $\mathcal{M}$-coded.*

Proof. We will argue in RCA_0. Say X is bounded by some $k \in \mathbb{N}$. By the primitive recursion of multiplication, we can define $m = k!$ as follows: define $f : \{x \in \mathbb{N} | x \leq k+1\} \to \mathbb{N}$, let $f(0) = 1$, and for $i < k$ let $f(i+1) = f(i) \cdot (i+1)$; then we let $m = f(k)$. By Σ^0_0 induction,

$$(\forall i < k)\ i+1 \text{ divides } m. \tag{1.25}$$

The claim is that for all $j < i < k, m(j+1)+1$ and $m(i+1)+1$ are relatively prime, that is, they do not divide each other. Say, for the sake of contradiction, that d divides both $m(j+1)+1$ and $m(i+1)+1$. So we set $m(j+1)+1 = dq_j$, and $m(i+1)+1 = dq_i$, and we see that their difference, $d(q_j - q_i)$ is also divisible by d, so d divides $m(i-j) := dq$ We then see that $m(j+1)+1$ divides $dq_jq = m(i-j)q_j$ Clearly m and $m(j+1)+1$ are relatively prime because of the last $+1$ term, so $m(j+1)+1$ must divide $(i-j)q_j$. However, $1 < i-j < k$, so $i-j$ divides m by (1.25). Thus, $m(j+1)+1$ must divide q_j. So since q_j divides $m(j+1)+1$, we have that $q_j = dq_jr$, $1 = dr$, so $d = 1$ which proves the claim.

By primitive recursion, define $n = \Pi_{i \in X}(m(i+1)+1$ as follows: let $g : \{x \in \mathbb{N} | x \leq k+1\} \to \mathbb{N}$, and set $g(0) = 1$. For $i < k$, let

$$g(i+1) = \begin{cases} g(i) \cdot (m(i+1)+1) & \text{if } i \in X \\ g(i) & \text{if } x \notin X \end{cases}$$

And set $g(k) = n$. Thus, $m(i+1)+1$ will divide n if $i \in X$. Induction on $l \leq k$ shows that the prime factors of $g(l)$ must be a factor of $m(i+1)+1$ for some $i < l$ in X, since $g(l)$ is nothing more than a sequence of products of $m(i+1)+1$ running i through up to l. By the above claim, $m(i+1)+1$ dividing $n = g(k)$ implies that it belongs in the sequence of products as a prime factor, and since it must be a factor of one of the $m(j+1)+1$ where $j < k$, but where $j \neq i$ these are relatively prime, so this implies that $m(i+1)+1$ is in the sequence, for some $j < k$, j = i, so $i \in X$.

We conclude that $r = (k, (m, n)$ represents X in the sense of Definition 1.22. The set R of all r representatives exists by Σ^0_0 comprehension, and it is nonempty because it contains r. If we let $\phi(x)$ be the Σ^0_0 formula $(\forall y \leq x)[y \notin R]$. We have

$\neg\phi(r)$, by Σ^0_0 induction there exists a $c \leq r$ such that $\neg\phi(c)$ and either $c = 0$ or $c > 0$ and $\phi(c-1)$, this implies that c is the least member of R, and that c is a code of X in the sense we desire. □

We finally relate finiteness and boundedness in models of RCA_0

Definition 1.26. For a model $\mathcal{M}$ of RCA_0, a set $X \subseteq M$ is $\mathcal{M}$-finite if it is $\mathcal{M}$-coded, $X \in S$, and $\mathcal{M}$-bounded

There is a fine distinction to be made between $\mathcal{M}$-finite sets and finite sets in the traditional sense when nonstandard first-order elements are present in the model $\mathcal{M}$. For example, in a nonstandard model $\mathcal{M}$, with nonstandard number $n \in M$, the set $N = \{i : i \leq n\}$ is $\mathcal{M}$-bounded by n in S (because we construct it with only bounded quantifiers), and is thus coded by some $c \in M$, and so is $\mathcal{M}$ finite. However, because n is nonstandard, for all $m \in \mathbb{N}$, $m < n$, so $\mathbb{N} \subset N$, yet is still $\mathcal{M} - finite$ when viewed from inside the model.

When considering standard models of RCA_0, which we will assume we have from now on, the traditional notion of finite and $\mathcal{M}$-finite coincide.

1.3. Proofs in RCA_0.

The first batch of things we will prove in RCA_0 deal with bounding, and that an additional type of useful comprehension holds in RCA_0.

Definition 1.27. Let Γ be a collection of formulas. The Γ bounding scheme ($B\Gamma$) is the scheme consisting of all $\phi(x, y) \in \Gamma$ of sentences of the form:

$$(\forall z)[(\forall x < z)(\exists y)\phi(x, y) \rightarrow (\exists w)(\forall x < z)(\exists y < w)\phi(x, y)]$$

It is quick to see that RCA_0 proves $\mathrm{B}\Sigma^0_0$ through Σ^0_0 comprehension. $\phi(x, y)$ be a Σ^0_0. Suppose that for some z we have that $(\forall x < z)(\exists y)\phi(x, y)$. Let $\theta(v)$ be the statement $v > z$ or $(\exists w)(\forall x < v)(\exists y < w)\phi(x, y)$. This is a Σ^0_1 statement, so we can induct on it for all v, and we have $\theta(z)$, which is the bounding statement.

We will now prove some things about general bounding schemes in PA^- that hold in the more specific cases pertaining to the allowed induction and comprehension schemes in RCA_0.

Theorem 1.28. *Fix $n \geq 0$, let t be a first order term.*

1. *If $\phi(x, z)$ is a Σ^0_n formula, then $(\forall x < t)\phi(x, z)$ is equivalent over PA^-+$B\Sigma^0_n$ to a Σ^0_n formula.*
2. *If $\phi(x, z)$ is a Π^0_n formula, then $(\exists x < t)\phi(x, z)$ is equivalent over PA^-+$B\Pi^0_n$ to a Π^0_n formula.*

Proof. We prove by induction on n. For $n = 0$, we know that Σ^0_0 formulae are closed under bounded quantification. Fixing $n > 0$, we assume the result for $n - 1$. We will prove (1), where (2) is analogous just by replacing the Σs with Πs and the $\forall$s with $\exists$s. Say $\phi(x, z) = (\exists y)\psi(x, y, z)$, where ψ is Π^0_{n-1}. In PA^-+$\mathrm{B}\Sigma^0_n$, we have

$$(\forall x < t)\phi(x, z) \leftrightarrow (\forall x < t)(\exists y)\psi(x, y, z) \leftrightarrow (\exists w)(\forall x < t)(\exists y < w)\psi(x, y, z)$$

. We apply the inductive hypothesis to $(\exists y < w)\psi(x, y, z)$, implying that it is equivalent to a Π^0_{n-1} formula $\upsilon(w, x, z)$, so $(\forall x < t)\phi(x, y)$ is equivalent to the Σ^0_n formula $(\exists w)(\forall x < t)\upsilon(w, x, z)$. This formula is Σ^0_n because the bounded universal is inside the scope of the existential. □

Theorem 1.29. *The following are provable in PA^-.*

(1) $B\Sigma^0_{n+1}$ *is equivalent to* $B\Pi^0_n$
(2) $I\Sigma^0_n$ + $B\Sigma^0_0 \to B\Sigma^0_n$

Proof. (1) The implication $B\Sigma^0_{n+1} \to B\Pi^0_n$ is immediate, since Π^0_n formulas are a subclass of $B\Sigma^0_{n+1}$. For the converse implication, consider a given Σ^0_{n+1} formula $\phi(x, y) = (\exists u)(\psi(x, y, u))$ Where ψ is a Π^0_n formula. Fix a z, and suppose that $(\forall x < z)(\exists y)\phi(x, y)$ holds. Define $\theta(x, v)$ to be the formula

$$(\forall y, u \leq v)[v = \langle y, v\rangle \to \psi(x, y, u)]$$

which, since we only added bounded quantifiers, is still a Π^0_n formula. Then, because of what we suppose about ϕ, we have that for all $x < z$ we can find a y such that $\phi(x, y)$ holds, and therefore a u such that $\psi(x, y, u)$ holds. Once we've found our y and u we code them into a v in the RCA_0 effective method, and by consequence they will be less than or equal to v. Thus, we have that $(\forall x < z)(\exists v)(\theta(x, v)$, and by $\mathrm{B}\Pi^0_n$ we can fix a w such that $(\forall x < z)(\exists v < w)\theta(x, v)$. Finally, since $y \leq v < w$, and because of what θ states about ψ and therefore ϕ, we have that $(\exists w)(\forall x < z)(\exists y < w)\phi(x, y)$.

(2) We prove by induction on n. For $n = 0$, there is nothing to show. Fixing $n > 0$, we assume the result for $n - 1$. We will argue in $\mathrm{PA}^-+\mathrm{I}\Sigma^0_n+\mathrm{B}\Sigma^0_0$, and by part (1), it suffices to show Π^0_{n-1}. Suppose $\phi(x, y)$ is a Π^0_{n-1} formula, such that for some z,

$$(\forall x < z)(\exists y)\phi(x, y).$$

Let $\psi(u)$ be the formula $u > z \vee (\exists w)(\forall x < u)(\exists y < w)\phi(x, y)$. By inductive hypothesis, we have $\mathrm{B}\Sigma^0_{n-1}$. We apply *Theorem* 1.28 to $(\exists y < w)\phi(x, y)$, implying it is Π^0_{n-1}, which implies Ψ is Σ^0_n with the extra existential quantifier. Thus, we will induct on ψ through $\mathrm{I}\Sigma^0_n$. $\psi(0)$ holds where there are no $x < 0$, so ψ holds vacuously. If we have $\psi(u)$, then either $u > z$, in which case $u + 1$ holds, or $u \leq z$ and $(\exists w)(\forall x < u)(\exists y < w)\phi(x, y)$. In this case, either $u + 1 > z$, and $\psi(u + 1)$ holds; or, $u + 1 \leq z$, in which case $(\forall x < z)(\exists y)\phi(x, y) \to (\forall x < u + 1)(\exists y)\phi(x, y)$ In particular, where $x = u$, we have that $(\exists y_u)\phi(u, y_u)$. Now we check if $y_u < w$, if yes, then keep w the same, if $y_u \geq w$, then redefine $w := y_u + 1$, and then you have, in either case, that $(\exists w)(\forall x < u + 1)(\exists y < w)\phi(x, y)$. This proves the inductive step. Thus, we have $(\forall u)\psi(u)$, and in particular, $\psi(z)$, which proves the desired claim. □

We quickly see a corollary to this theorem relevant to RCA_0, that:

Corollary 1.30. RCA_0 *proves* $B\Sigma^0_1$.

Proof. Let $n = 1$ in Theorem 1.29, RCA_0 has $\mathrm{B}\Sigma^0_0$ and $\mathrm{I}\Sigma^0_1$, so RCA_0 proves $\mathrm{B}\Sigma^0_1$. □

We will need one more preliminary result before we prove that an additional type of comprehension holds in RCA_0. This result regards a *least number principle* in RCA_0.

Definition 1.31. Let Γ be a collection of formulas. The Γ least number principle ($\mathrm{L}\Gamma$) is the scheme over all $\phi(x) \in \Gamma$ of sentences of the form

$$(\exists x)\phi(x) \to (\exists x)[\phi(x) \wedge \neg(\exists y < x)\phi(y)].$$

Theorem 1.32. *For* $n \geq 1$, PA^- *proves that* $I\Sigma^0_n \to L\Pi^0_n$

Proof. Let $\phi(x)$ be a Π^0_n formula, and define $\psi(x)$ to be the formula $(\forall y \leq x)\neg\phi(y)$, since the negation of a Π^0_n formula is Σ^0_n, and we are working in the naturals, we can equivalently write $\psi(x) = (\forall y < x+1)\neg\phi(y)$, by *Theorem* 1.28 $\psi(x)$ is Σ^0_n. We assume for the sake of contradiction that we do not have $L\Pi^0_n$, we will induct on $\psi(x)$ using $I\Sigma^0_1$. $\psi(0)$ holds where there is no least element satisfying ϕ, since $(\forall y \leq 0)\neg\phi(x)$, as we have $\neg\phi(0)$, or else ϕ would have a least element. Assuming $\psi(x)$ holds, $\psi(x+1)$ must also hold, or else $x+1$ would be the least element satisfying ϕ. Thus, by Σ^0_n induction, $(\forall x)\psi(x)$, so $(\forall x)\neg\phi(x)$, this is a contradiction for any Π^0_n sentence that hold for some x, which exist. ☐

Similarly, we see a quick application to RCA_0.

Corollary 1.33. *RCA_0 proves $L\Pi^0_1$*

Proof. Let $n = 1$ in Theorem 1.32, RCA_0 has $I\Sigma^0_1$, so RCA_0 proves $L\Pi^0_1$ ☐

We now have all we need to prove the result that, in similar fashion to the last two theorems, will apply to RCA_0 with little friction and reveal an important comprehension scheme that holds in RCA_0.

Theorem 1.34. *Fix $n \geq 1$, and let $\mathcal{M}$ be a model that supports RCA_0. The following are equivalent.*

(1) $\mathcal{M}$ supports $I\Sigma^0_n$
(2) Every Σ^0_n definable, bounded subset of M is $\mathcal{M}$-finite.

Proof. ($1 \to 2$) Let $\phi(x)$ be a Σ^0_n formula, and fix $k \in M$. Let $m = k!$ as defined in Theorem 1.24, so we know that m is recursively defined. Consider the following formula in which n is a free variable:

$$(\forall i < k)[\phi(i) \to m(i+1)+1 \text{ divides } n \tag{1.35}$$

The above holds where $n = \Pi_{i<k} m(i+1)+1$. By $B\Sigma^0_n$ in $\mathcal{M}$ (Theorem 1.29), the above formula is Π^0_n. Then, by Theorem 1.32 we have $L\Pi^0_n$, so there is a least $n \in M$ satisfying (1.35). Suppose there is an $i <_{\mathcal{M}} k$ such that $m(i+1)+1$ divides n but $\neg\phi(i)$. We write $n = (m(i+1)+1)n^*$, from the information that $m(i+1)+1$ divides n, for some $n^* <_{\mathcal{M}} n$, the inequality is strict because the minimum value of $(m(i+1)+1)$ is 2. By Theorem 1.24, $\mathcal{M}$ satisfies

$$(\forall j < k)[j \neq i \to m(j+1)+1 \text{ and } m(i+1)+1 \text{ are relatively prime.}$$

This however implies that n^* satisfies (1.35), since we know that when we consider all $j < k$ for that formula, we need not consider the i such that $\neg\phi(i)$, and every other $m(j+1)+1$ divides n^*, but n was supposed to be the least element that satisfies the formula. So we have a contradiction. Thus, (1.35) must actually be a biconditional so we have that $\mathcal{M}$ satisfies that $\langle k, \langle m, n\rangle$ represents the set $\{i < k : \phi(i)\}$ in the desired sense of Definition 1.22, so the set is $\mathcal{M}$-finite.

($2 \to 1$) Let $\phi(x)$ be a Σ^0_n formula such that $\mathcal{M}$ proves that $\phi(0) \wedge (\forall x)[\phi(x) \to \phi(x+1)]$ Fix an arbitrary $a \in M$. By the assumed (2), the set F_a of $b \in M$ such that $b < a+1 \wedge \phi(b)$ is $\mathcal{M}$-finite, since it is Σ^0_n definable. We can thus use a code for F_a, and express the formula $x \in F_a$ as a Σ^0_0 formula, since checking if something is in a coded set against the code is a Σ^0_0 formula as previously discussed. Thus, we can induct on the membership of the set, like $\psi(x)$ being the Σ^0_0 formula $x > a \vee x \in F_a$. Then we have that $\psi(0) \wedge (\forall x)[\psi(x) \to \psi(x+1)]$. By $I\Sigma^0_0$ in RCA_0, we can induct on ψ and conclude that $\psi(a)$ holds, so $a \in F_a$, so $\phi(a)$, since a was arbitrary, we have proved $(\forall a)\phi(a)$ and proved $\mathrm{I}\Sigma^0_n$. □

We now use the above theorem for a quick proof of Bounded Σ^0_1 comprehension.

Theorem 1.36. *For every Σ^0_1 formula $\phi(x)$, RCA_0 proves that*

$$(\forall z)(\exists X)(\forall x)[x \in X \leftrightarrow x < z \wedge \phi(x)]$$

Proof. We follow the same tack as the second part of the proof of Theorem 1.34. Fixing an $a \in M$, we can code the set $F = \{x < a : \phi(x)\}$ with a code $c \in M$. We can now just check membership in this set F with a Σ^0_0 formula, and then form it with Σ^0_0 comprehension. □

Note that if we add higher forms of induction, $\mathrm{I}\Sigma^0_n$ for $n > 1$ to RCA_0, we get a similar bounded Σ^0_n comprehension in exchange.

We will now prove that a known computability-theoretic fact about Σ^0_1 sets is provable in RCA_0, namely that they are the range of a computable function. We will need a few preliminary results first.

Theorem 1.37. *In RCA_0, for an infinite set S, there exists a function p_S called the principal function of S, such that $range(f) = S$, and for $n < m$, $p_S(n) < p_S(m)$.*

Proof. We can order our set S in increasing order, and then define the function computibly for each input x, such that on input x we scan the ordered S starting from the bottom until we've reached the xth element of S ordered sequentially. This gives a finite algorithm for determining the pairs of the function p_S, so p_S is computable from S, and RCA_0 proves computibly definable sets exist. (This is a somewhat informal argument, and relies on the fact that recursive constructions, like checking against the ordering of S recursively, are justified by Δ^0_1 comprehension and Σ^0_1 induction.) □

Theorem 1.38. *(1) is equivalent to (2) in RCA_0, given a model $\mathcal{M}$ of RCA_0, for a function $f : M \to M$:*

(1) f is Σ^0_1 definable.
(2) f is Δ^0_1 definable.

Proof. We will prove the theorem by means of computability theory. If a function f is *c.e.*, since a function is nothing more than a coded set $\{\langle x, y\rangle : f(x) = y\}$ We will prove for a *c.e.* graph, which is what f being *c.e.* implies, that this implies that the function is computable. Given an input n into f, we know that a computable function g outputs one pair (n, y) in finite time, since the graph is *c.e.*, so just wait for g to output (n, y) and set $f(n) := y$. This proves f is computable, and therefore Δ^0_1 □

We have the preliminary results to prove the final theorem of this section.

Theorem 1.39. *Let ϕ be a Σ^0_1 formula. Then RCA_0 proves there is a function $f : \mathbb{N} \to \mathbb{N}$ such that for every y, $\phi(y)$ holds iff $y \in range(f)$. Moreover, if $\phi(y)$ holds for infinitely many y then f may be chosen to be injective.*

Proof. We will begin with the finite case. Suppose there is a finite set F such that for all y, $y \in F$ iff $\phi(y)$ holds. In this case, we define f by $f(y) = y$ for all $y \in F$ and $f(y) = inf(F)$ for all $y \notin F$

In the case that no such F exists, write $\phi(y)$ as $(\exists x)\psi(x, y)$, where ψ is Σ^0_0. Let S be the set of pairs $\langle x, y\rangle$ such that

$$\psi(x, y) \wedge (\forall x^* < x)\neg\psi(x^*, y).$$

By $\mathrm{L}\Pi^0_0$ in RCA_0, we have that $\phi(y) \leftrightarrow (\exists x)[\langle x, y\rangle \in S]$. If S were bounded by some b, we would have that this $x < b$, or that $\phi(y) \leftrightarrow (\exists x < b)[\langle x, y\rangle \in S]$. Thus the set of y such that $\phi(y)$ would exist by Σ^0_0 comprehension, and would be finite, which contradicts our assumption that no such F exists. Thus S is unbounded. Let p_S be the principal function of S as defined in Theorem 1.37, we know p_S is injective with range S. Define f by

$$\langle z, y\rangle \in f \leftrightarrow (\exists x)[\langle z, x, y\rangle \in p_S].$$

f is a function because p_S is, and is injective because if $\langle x, y\rangle, \langle x^*, y\rangle \in S$, and $x \neq x*$, we have that $x^* < x$ (other case is symmetric), so $\neg\psi(x^*, y)$ by the definition of S, which contradicts $\langle x^*, y\rangle \in S$, so $x^* = x$. Thus f is injective. Since f is Σ^0_1 definable, by Theorem 1.38, it is Δ^0_1 definable, so f exists in the RCA_0 sense, and f is the function we desired. □

Corollary 1.40. *The above f is monotonic increasing.*

Proof. The Cantor pairing function is monotonic increasing w.r.t. each argument, so p_s's monotonicity implies that f is monotonic increasing, as it is the composition of these two. □

2. Trees, Weak Kőnig's Lemma, and WKL_0

To motivate our next system, WKL_0, we need to introduce trees and relate them to our existing discussion of computable mathematics.

Definition 2.1. A tree is a subset T of $\mathbb{N}^{<\mathbb{N}}$ (finite length strings over $\mathbb{N}$) such that if $\sigma \in T$, then every initial segment of σ is also in T. A tree T is finitely branching if for every $\sigma \in T$, there are only finitely many extensions of σ that are still in T. (We denote an extension like σn, so there are only finitely many n such that $\sigma n \in T$.) A tree is binary if it is a subset of $2^{<\mathbb{N}}$. A path on a tree T is an $X \in \mathbb{N}^{<\mathbb{N}}$ such that for all n, $X \uparrow n \in T$ (the initial segment of X up to the *nth* position in the string.)

A familiar term is used to denote the set of all all paths on a tree T, denoted as $[T]$.

Definition 2.2. A Π^0_1 class is a set of the form $[T]$ for some computable binary tree T

The familiar usage of the term Π^0_1 hints at a connection to the Π^0_1 form of a set, as this is actually the case.

Theorem 2.3. *P is of the form $[T]$ for a computable binary tree iff P is Π^0_1 definable.*

Proof. We will begin with the forwards implication. For $P = [T]$, the set T is a computable binary tree, so if $X \in P$, X is some path on T, which implies, since T is computable, that there exists a set $S := T$ such that $X \in P$ iff $(\forall n)X \uparrow n \in S$, which is a definition in Π^0_1 form. For if $X \in P$ then by definition $\forall n X \uparrow n \in S$, since X is a path on T, and the converse also holds since P is of the form $[T]$, and the statement $\forall n X \uparrow n \in S$ implies X is a path on T.

The converse states that for a class of sets P, if P is defined in a Π^0_1 form, then P is of the form $[T]$ for a computable binary tree T. To see this, we can expose an implication of our definition of a Π^0_1 form for a function in the following way. Originally, Π^0_1 form was $\forall x \phi(X, x)$, where ϕ is Σ^0_0, and X is a free set variable. However, a Σ^0_0 formula is also Π^0_1 and Σ^0_1 with the addition of dummy quantifiers, so all Σ^0_0 formulas are computable. Thus, if ψ is Π^0_1, then it is of the form $\forall x \phi(X, x)$, where ϕ is computable, and X can be thought of as an oracle on the computable function with input x. Computable functions are equivalent to a binary computation tree T, that consist of strings σ such that if $\phi(\sigma, n)$ converges in $|\sigma|$ steps, then it outputs 1. Here, the oracle set X has been coded into a finite string oracle σ, and if ϕ attempts to query outside the length of σ, the computation is undefined. So, interpreting each x as a depth on the decision tree, $\forall x$ implies that σ is a path through said binary computation tree. The collection of all σ such that $\forall x \phi(\sigma, x)$ is therefore of the form $[T]$.

□

Thus we have that $[T]$ for a computable binary tree, or anything that matches that kind of form, can be defined in the Π^0_1 formula specified in the proof. Thus, since it has that form of definition, it is a set in Π^0_1 form.

We now come onto the principle that gives WKL_0 its name, Weak Kőnig's Lemma. We will give a general proof of the theorem, not restricting our axioms.

Theorem 2.4. *(Weak Kőnig's Lemma) Every infinite binary tree has a path.*

Proof. For a general infinite binary tree T, start at the empty string $\emptyset$, and make your choice on which node to follow in your path by asking the question "is there an infinite number of nodes following my choice?", because the tree is infinite, one of the two options will result in an infinite subtree that begins at the node of your choosing. We follow this construction inductively, the inductive step being fairly clear just by replacing the empty string with whatever string we've ended up on after k steps, and then choosing the $k + 1$th node based on our algorithm. □

There should immediately be some alarms going off that this kind of construction is not effective, as the membership of the set R, the infinite path, is decided for each node by a non finitary question, that is, is there an infinite number of nodes above the one of our choosing? This would be a correct suspicion, as we can show that WKL_0 is strictly more powerful than RCA_0 with the following construction.

Definition 2.5. For A, B, disjoint sets, a separating set X is a set such that $A \subseteq X$ and $X \cap B = \emptyset$.

Theorem 2.6. *If A and B are disjoint c.e. sets, then the class of separating sets $\mathcal{X}$ is a Π^0_1 class.*

Proof. We would like to find a computable relation R such that $(\forall n)(R(C, n))$ holds iff R separates A and B. This will show that the class of seprating sets is a Π^0_1 class based on the definition laid out in *Theorem* 2.3. Consider the computable relation $R := (\forall i < n)(($if i enters A at stage $< n$, then $C(i) = 1.)$ and (if i enters B at stage $< n$, then $C(i) = 0))$. This is coding A and B into trees and n and is into strings and looking at when, exactly, i goes from being not in A or

B to in A or B. C will then look like a separator of A and B up to stage n on inputs $< n$. Running through R $\forall n$ thus results in a seperator R and we have the condition we were looking for. □

Theorem 2.7. *There are c.e. disjoint sets A, B with no computable separation.*

Proof. Consider the sets $A = \{x|\phi_x(0) = 0\}$ and $B = \{x|\phi_x(0) = 1\}$. Assume the existence of a computable separator C for the sake of contradiction. Kleene's Recursion Theorem, a famous and fundamental theorem from computability theory, states that for a partial computable function f, we can find some n such that $\phi_n(j) = f(\langle n, j\rangle)$. Since C is computable, its characteristic function is a computable function, call it g. Thus, we can find an n such that $\phi_n(0)$ checks for n's membership in C. If yes, then output 1, if no, then output 0. Thus, we have that $C(n) = \phi_n(0) = 0$ implies $n \in A$, $C(n) = \phi_n(0) = 1$ implies $n \in B$, but C was a separator of A and B, so this is a contradiction, since this implies it shares n with B, or $n \in A$ but the characteristic function for C outputs 0, so $n \in A$ but $n \notin C$, which is also a contradiction to C being a seperator. □

Corollary 2.8. *There are computable binary trees with no computable paths.*

Proof. Theorem 2.6 together with Theorem 2.7 states that a computable tree T exists such that no member of $[T]$ is computable. □

This finishes the separation between RCA_0 and WKL_0, as Weak Konig's Lemma guarantees the ability to always find a path no matter the infinite binary tree T, but RCA_0 gets stuck on the above tree implied by *Corollary* 2.8. The separation is given by the model $\mathcal{M}$ that has as its second order part an ideal of all computable sets. This model includes all computable binary trees, but by *Corollary* 2.8 does not include all of their paths. Thus, since the existence of paths through all computable binary trees is specified by WKL_0, we say that this model models RCA_0, yet not WKL_0. We can be more specific about just how much further above WKL_0 is from RCA_0 in terms of computability-theoretic strength by looking at the concept of the Turing Jump Operator, and the practice of stratifying this notion of strength through the Turing Degrees.

Definition 2.9. The Turing Jump Operator is defined by its action on a set X. For a set X, its Turing Jump is defined as $X' = \{x|\phi_x^X(x) \text{ halts.}\}$ In words, it is the set defined by the points x such that the action of the Turing machine with oracle X, label x, halts on input x.

Definition 2.10. Turing reducibility is a relationship defined between two sets A and B such that A is Turing reducible to B if for a point $a \in \omega$, a's membership in A can be decided by an oracle Turing machine ϕ with oracle B. In this case, we write $A \leq_T B$

Definition 2.11. A Turing degree is an equivalence class defined by the following relation, if A and B are Turing reducible to each-other, then A and B are said to have the same Turing degree, we write $A =_T B$, and denote the degree by $\mathbf{A}$, or any other bolded representative element (in the case of the $\emptyset$'s Turing degree, we often write $\mathbf{0}$).

Thus the aforementioned ideal of all computable sets is an ideal $I = \{X|X \leq_T \mathbf{0}\}$

There exists a special kind of set we can specify using these concepts which stratify computational strength and difficulty, called a *low* set.

Definition 2.12. A set X is *low* if $X' =_T \mathbf{0}'$, where $\mathbf{0}'$ is the degree of the Turing Jump of the empty set.

Theorem 2.13 (Low Basis Theorem)**.** *Every nonempty* Π^0_1 *class has a low member.*

Proof. Let T be a computable binary tree, we wish to construct a low path $R \in [T]$. Since Π^0_1 classes are of the form $[T]$, showing that there always exists a low path in $[T]$ for an arbitrary computable tree T will suffice to prove the theorem.

This construction will be $\mathbf{0}'$ effective, and we will obtain all digits of the jump of our path through using only $\mathbf{0}'$, ensuring that the path is low.

At stage n I will have a string σ_n such that

$$\emptyset = \sigma_0 < \sigma_1 < ... < \sigma_{n-1} < \sigma_n < ... = \bigcup_{n \in \mathbb{N}} \sigma_n =: \pi$$

Such that each σ_i extends σ_{i-1}. Ultimately, the string π will be the low string we are looking for through the construction of each σ_n. I will also define a subtree $T_n \subset T$ such that every path through T_n extends σ_n, this is a 'tree of possibilities' from σ_n: a subtree of all of its possible extensions in T.

At even stages, $e \in \mathbb{N}_0$, $2e$, I will lengthen σ_n to find a τ that extends σ_n by one digit such that there are infinitely many nodes above τ in T_n. I can do this with the computational power of $\mathbf{0}'$ in the following way. The question: "are there infinitely many nodes above τ?" can be rephrased in the following way: for every n-length extension of the path τ, such that τ_1 is any path in T that extends τ by one digit, and τ_n is an n-length extension of τ in T, does there always exist a τ_n for all n? We can rephrase this question further into an algorithm, ϕ. ϕ on input k considers the finite k-length extensions τ_k and asks if any of them are in T. For each finite value n this is a computable algorithm, since we only have a finite computable (since T is) subtree to check. It halts whenever it finds this τ_k on input k, we don't care about its output, just set it to 1. Since ϕ is computable, as it runs a finite algorithm for each n, the question on whether is halts or not is $\mathbf{0}'$-effective. We know ahead of time that for some option τ, the answer is no, because of WKL. We choose τ where ϕ doesn't halt. So our choice of τ, and assigning $\tau := \sigma_{n+1}$ at even stages results in an infinite path.

At odd stages $2e+1$ consider the set $U_e = \{\tau_n \in T_n | \Phi^{\tau}_e(e)$ does not halt in $|\tau_n|$ steps$\}$ and ask: is this set infinite? Where τ_n are $n \in \mathbb{N}$ length strings in T_{n+1} as defined in the previous even step. Again, $\mathbf{0}'$ can answer this. Consider the two variable Turing function $\phi_e(\tau_n, s) := \Phi^{\tau_n}_{e,s}(e)$, which is running the Turing functional with oracle τ_n and running through s from 1 to $|\tau_n|$ as steps in the Turing function, once it does this, if it still hasn't halted anywhere, we have our answer that τ_n should be in the set, and if it has, then τ_n should not be in the set. Since this is a question about the halting of a finitary Turing program, $\mathbf{0}'$ can answer it. Setting a finite number of steps makes sure we are never actually asking questions about the jump of any set but $\mathbf{0}$, since we will stop computing when we get to $|\tau_n|$ many steps no matter what. However because of WKL and our choice of T_n, we know that no matter what n is, we can always find a τ_n as we know an infinite string exists in T_n.

Now, we can talk about what these answers imply. If yes, that U_e is infinite, we assign $T_{n+2} = U_e$, and we have determined that for this e, since we can follow an infinite path through U_e as it is itself infinite, that $\Phi_e^\pi(e)\uparrow$. This implies that $e \notin \pi'$. If no, and U_e is finite, we won't touch T_{n+1}, and let $T_n = T_{n+2}$, and then we know that eventually, the infinite path we know exists through T_n will leave U_e, so $\Phi_e^\pi(e)\downarrow$. This implies that $e \in \pi'$

We have fully determined the members of π' using only the power of $\mathbf{0}'$, so we know π is low. □

We remark that this fact, combined with the fact that there exist trees with no computable path, imply that there exists a path of said tree which is low yet non-computable. This fact adds sets that "fill the gap" between RCA_0 and WKL_0.

Before we move on to our next system, ACA_0, we'll introduce some of the work that is done with all of this machinery with an example proof of a well-known mathematical fact that only needs the power of WKL_0 to prove, as well as implies WKL_0. This 'implication of axioms' half is exactly where the name 'reverse mathematics' comes from, instead of solely working from the axioms, we see which axioms are necessary and sufficient for a given mathematical principle. This allows us to determine a 'spectrum of strength' across mathematics, where this 'strength' is its location in the arithmetic hierarchy. We will be doing this with Heine-Borel as a good example case.

Theorem 2.14. *The following is provable in WKL_0. Given sequences of real numbers $(c_i)_{i\in\mathbb{N}}$, $(d_i)_{i\in\mathbb{N}}$, if*

$$\forall x(0 \le x \le 1 \to \exists i(c_i < x < d_i))$$

then,

$$\exists n \forall x(0 \le x \le 1 \to \exists i \le n(c_i < x < d_i))$$

This corresponds to the traditional notion of Heine-Borel wherein one proves that every open cover of $[0,1]$ has a finite subcover. Where the c_is and d_is are the endpoints of the intervals forming said open cover.

Proof. We will first prove the claim for sequence of rational numbers, and extend the claim to reals with a later argument. For each string $s \in 2^{<\mathbb{N}}$ put

$$a_s = \sum_{i<length(s)} \frac{s(i)}{2^{i+1}}$$

and

$$b_s = a_s + \frac{1}{2^{length(s)}}.$$

Thus, for each $n \in \mathbb{N}$ we have partitioned the unit interval $0 \le x \le 1$ into 2^n sub-intervals of length 2^{-n}, namely, $a_s \le x \le b_s$, where $length(s) = n$, as for a given n there are 2^n different finite binary sequences of length n. Form a tree $T \subseteq 2^{<\mathbb{N}}$ by putting $s \in T$ iff $(\nexists i \le length(s))(c_i < a_s < b_s < d_i)$ T exists by Σ_0^0 comprehension since $c_i, d_i, a_s, b_s \in \mathbb{Q}$. T is also clearly a tree since membership of $s \in T$ implies all the initial segments of s are also in T, as well as nonmembership in T of any s implying that no initial segments of s are in T.

Assuming that $\forall x(0 \le x \le 1 \to \exists i(c_i < x < d_i))$, we claim that T has no path. To see this, let $f \in 2^{\mathbb{N}}$ and $f : \mathbb{N} \to \{1, 0\}$, put x such that

$$x = \sum_{j=0}^{\infty} \frac{f(j)}{2^{j+1}}$$

i.e. the unique x such that $a_{f[n]} \leq x \leq b_{f[n]}$ where $f[n]$ is defined as the sequence $(f(1), f(2), ..., f(n))$ for all $n \in \mathbb{N}$. We see that x is unique in this role in the infinite limit as $n \to \infty$ and $b_{f[n]} \to a_{f[n]}$. Let i be such that $c_i < x < d_i$ and let n be so large that $n \geq i$ and $c_i < a_{f[n]} < b_{f[n]} < d_i$. And then $f[n] \notin T$, since the function was arbitrary, this proves the claim.

By weak Konig's lemma it follows that T is finite. Let n be such that $\forall s(s \in T \to length(s) < n)$. Then $\forall s(length(s) = n \to \exists i \leq n(c_i < a_s < b_s < d_i))$. Hence $\forall x(0 \leq x \leq 1 \to \exists i \leq n(c_i < x < d_i))$.

We have proved the theorem under the restraints that $c_i, d_i \in \mathbb{Q}$. Expanding this to $\mathbb{R}$, we consider the Σ^0_1 formula $\phi(q, r)$ which says that $q \in \mathbb{Q} \wedge r \in \mathbb{Q} \wedge \exists i(c_i < q < r < d_i)$. By Theorem 1.39 there exists a function $f : \mathbb{N} \to \mathbb{Q} \times \mathbb{Q}$ such that $\forall q \forall r(\phi(q, r) \leftrightarrow \exists j(f(j) = (q, r)))$, we can fit f to the conditions of Theorem 1.39 as we can find a bijection between $\mathbb{Q} \times \mathbb{Q}$ and $\mathbb{N}$ since the space is countable. We thus replace the sequence $((c_i, d_i) : i \in \mathbb{N})$ by the sequence $\langle q_j, r_j \rangle : j \in \mathbb{N}\rangle$ where $(q_j, r_j) = f(j)$. This reduces to the special case which has already been proved. □

We end our discussion of WKL_0 with a taste of reverse mathematics. That being reversing the implication in the above proof. Is Heine-Borel sufficient to prove the axioms of WKL_0? This provides a definitional link between Heine-Borel and WKL_0, and stratifies Heine-Borel along the computability-theoretic strength hierarchy at WKL_0. We are gauging the strength of our mathematics and quantifying the idea of, say, the phrase mathematicians like to use when 'complex' mathematics are used to prove a theorem when 'simpler' mathematics could also be used–'heavy' vs. 'light' machinery.

Theorem 2.15. *WKL_0 is equivalent to Heine-Borel over* $[0, 1]$.

Proof. We will reason in RCA_0 as our base system. Assume Heine-Borel, i.e. the result of the previous theorem. Consider the Cantor middle-thirds set $C \subseteq [0, 1]$ which consists of all real numbers of the form:

$$\sum_{i=0}^{\infty} \frac{2f(i)}{3^{i+1}}, f \in 2^{\mathbb{N}}.$$

Where $f \in 2^{\mathbb{N}}$ means that $f = \{(\langle i, s\rangle)^{i\in\mathbb{N}}_{s\in S} | S \in 2^{\mathbb{N}}_0\}$ where $f(i) = (i + 1)$th element in the binary string S, since our enumeration starts at 0.

The proof structure will be that the paths through $2^{<\mathbb{N}}$ can be identified with elements of C, so the Heine-Borel compactness of $2^{\mathbb{N}}$ follows from the Heine/Borel compactness of the unit interval $0 \leq x \leq 1$. By utilizing a compactness argument, we can move from infinite binary strings to finite binary strings. For each $S \in 2^{<\mathbb{N}}$ put

$$a_S = \sum_{i<length(S)} \frac{2S(i)}{3^{i+1}}$$

and

$$b_S = a_S + \frac{1}{3^{length(S)}}$$

Thus, we can see that for the empty string $\langle\rangle$, we see that $a_{\langle\rangle} = 0$, and $b_{\langle\rangle} = 1$, and for the general closed interval $a_S \leq x \leq b_S$, $a_{S\frown\langle 0\rangle} \leq x \leq b_{S\frown\langle 0\rangle}$ is the left third of that interval, and $a_{S\frown\langle 1\rangle} \leq x \leq b_{S\frown\langle 1\rangle}$ is the right third. Thus, for any $x \in C$, there exists a unique $f : \mathbb{N} \to \{0, 1\}$, such that for some n, the binary string $f[n] = (f(1), f(2), ..., f(n))$, we have $a_{f[n]} < x < b_{f[n]}$. Since we can specify any interval we would like in the Cantor middle-thirds set construction given infinite precision up to $n \in \mathbb{N}$. If $x \notin C$ then at some finite point in the Cantor middle-thirds set construction we 'throw away' this point, so we know there is a unique $S \in 2^{<\mathbb{N}}$ such that $b_{S^\frown\langle 0\rangle} < x < a_{S^\frown\langle 1\rangle}$. We also put

$$a'_S = \sum_{i<length(S)} \frac{2S(i)}{3^{i+1}} - \frac{1}{3^{length(S)+1}} = a_S - \frac{1}{3^{length(S)+1}}$$

$$b'_S = b_S + \frac{1}{3^{length(S)+1}}$$

Note that the open intervals $a'_S < x < b'_S$ and $a'_R < x < b'_R$ are disjoint unless $S \subseteq T$ or $T \subseteq S$, where the subset notation denotes a sub-string. Let $T \subseteq 2^{<\mathbb{N}}$ be a tree with no path. We will use Heine Borel to prove that T is finite, therefore implying Weak Konig's Lemma. Let $\bar{T}$ be the set of $U \in 2^{<\mathbb{N}}$ such that $U \notin T \wedge \forall R(R \subset U \to R \in T)$. $\bar{T}$ is thus a pairwise disjoint collection of strings, and the set $\{U \in \bar{T} | (a'_U, b'_U)\}$ covers C, since we include all the endpoints by making each 'kept' interval wider. By Heine Borel, since $\{U \in \bar{T} | (a'_U, b'_U)\} \cup \{S \in 2^{<\mathbb{N}} | b_{S^\frown\langle 0\rangle} < x < a_{S^\frown\langle 1\rangle}\}$ cover $[0,1]$, we know there exists a finite subcover of this set. The removed middle thirds (the latter set) are disjoint from C, and no proper subset of the set $\{U \in \bar{T} | (a'_U, b'_U)\}$ covers C, this therefore implies that the set $\bar{T}$ itself must be finite, and therefore T as well. □

This concludes our discussion of WKL_0, and now we move on to ACA_0, an even more powerful system.

3. Arithmetic Comprehension and ACA_0

We first remind the reader of the definition of an arithmetic formula, that is, a formula which does not quantify over set variables. Thus, something like Weak Konig's Lemma, which quantifies over all binary trees, in other words, quantifying over sets, would not be an arithmetic formula.

ACA_0 is a more natural extension of RCA_0, in the sense that it just modifies the definition of RCA_0 in the following way.

Definition 3.1. ACA_0 is P_0 together with a general comprehension axiom such that for any arithmetic formula ϕ such that ϕ has no bound set variable X, $\forall x[\phi(x)] \to \exists X \forall x[x \in X \leftrightarrow \phi(x)]$.

This comprehension axiom is known as the *Arithmetic Comprehension Axiom*, the namesake of ACA_0.

While it may seem that we have made a large jump from RCA_0, which only allowed for very basic restricted comprehension, the two systems are closer than they appear.

Theorem 3.2. *The axioms of RCA_0 with Σ^0_1 comprehension are equivalent to ACA_0.*

Proof. Since each arithmetical formula is reducible to some Σ^0_k formula for some $k \in \omega$ by definition, it suffices to prove that Σ^0_1 comprehension implies Σ^0_k comprehension. We prove by induction on k. For $k \leq 1$ the implication is trivial. For

$k \geq 1$. let $\phi(n)$ be Σ^0_{k+1}. Write $\phi(n)$ as $\exists j \psi(n,j)$ where $\psi(n,j)$ is Π^0_k. By Σ^0_k comprehension let Y be the set of all (n,j) such that $\neg\psi(n,j)$ holds. Then by Σ^0_1 comprehension let X be the set of all n such that $\exists j((n,j) \notin Y)$. Thus, $n \in X$ iff $\exists j \psi(n,j)$, so $\phi(n)$, this completes the proof. □

Corollary 3.3. *The existence of a range set for injective functions $f : \mathbb{N} \to \mathbb{N}$ is equivalent to ACA_0*

Proof. Since ACA_0 implies Σ^0_1 comprehension, and we can phrase the existence of the range set as a Σ^0_1 comprehension sentence: there exists a set $X \subset \mathbb{N}$ such that $\forall n(n \in X \leftrightarrow \exists m(f(m) = n))$, we have the first direction. The opposite direction is obtained by considering Theorem 1.39, since we can re-express satisfaction of a Σ^0_1 formula by the membership in a range set of an injective function equivalently, the existence of a range set for every injective $\mathbb{N} \to \mathbb{N}$ function implies Σ^0_1 comprehension. □

Corollary 3.4. *For an omega model $\mathcal{M}$ that supports ACA_0, for all $n > 0$, the set $\mathbf{0}^{(n)} \in S$*

Proof. For each n, the set $\mathbf{0}^{(n)}$ can be defined arithmetically by iterating on the definition of the Turing jump. □

To separate WKL_0 from ACA_0, we construct a model that supports WKL_0 consisting entirely of low sets and therefore does not include $\mathbf{0}'$.

To do this, we need to slightly improve on our low basis theorem.

Theorem 3.5 (Improved Low Basis Theorem)**.** *If T is a low infinite binary tree, T has a path π such that $T \oplus \pi$ is low.*

Proof. The proof here is mechanically identical to the proof of the normal low basis theorem, and is left out for the sake of brevity and non-redundancy. □

Now, we have the tools we need to construct a Scott ideal of low sets, that is, a class of sets S which has the closure properties of the Turing ideal, along with an additional closure property that for an infinite tree $T \in S$, there is a path through T in S. These are the ideals which model WKL_0 by having Weak Konig's Lemma as its defining property. Scott Ideals play a similar role in the construction of ω-models that model WKL_0 to Turing ideals and ω-models of RCA_0. By constructing a Scott ideal with the property that all sets in the ideal are low, we prove that we can model WKL_0 using the low Scott ideal as the collection of sets S that exist in the model. Thus, providing a model which supports WKL_0 yet does not support ACA_0, since it only contains low sets.

Theorem 3.6. *There is a Scott ideal consisting entirely of low sets.*

Proof. We shall construct a sequence of low sets: $X_0 \leq_T X_1 \leq_T X_2 \leq_T ...$ and let $S = \{Y | \exists n\ Y \leq_T X_n\}$. Let $\Phi_0, \Phi_1, \Phi_2, ..., \Phi_e, ...$ be an enumeration of oracle Turing machines such that every oracle Turing machine occurs infinitely often. We begin the construction assigning $X_0 = \emptyset$, and considering the oracle Turing machine $\Phi_0^{X_0}$, we ask the question: Is this a low infinite binary tree? If no, let $X_1 = X_0$, if yes, it has a low path. Let X_1 be such a path $\oplus\ \Phi_0^{X_0}$. Induct on this strategy to X_n, and ask "is $\Phi_n^{X_n}$ a tree?", and proceed as before, joining the path and $\Phi_n^{X_n}$ to form X_{n+1} when the answer is yes, so as the improved low basis theorem, states, each X_n will be low and increasing in the Turing degrees. We have thus constructed a Scott ideal consisting of low sets, S and proved the distinction of WKL_0 and ACA_0, since we include a path of all the infinite binary trees that are included in the Scott ideal, and can properly support WKL_0, while not supporting ACA_0. □

We end this section, and this paper, with a similar preview of reverse mathematics as in the WKL_0 section:

Theorem 3.7. *ACA_0 is equivalent to the principle that 'every countable vector space over a countable field has a basis.'*

Proof. We begin by proving that ACA_0 implies this principle. We reason in ACA_0. Let V be a countable vector space over a countable field K. By Arithmetic Comprehension, there exists an S consisting of all finite sequences $\langle v_0, ...v_{n-1}, v_n\rangle$, $n \in \mathbb{N}$, such that $v_n = \sum_{i<n} a_i \cdot v_i$ for some $a_0, ..., a_{n-1} \in K$ Using S as a parameter, we apply primitive recursion (a simple kind of finitely halting algorithm supported by RCA_0 in the context of computability theory) to define a sequence of vectors $e_0, e_1, ..., e_n, ...$ where $e_n =$ the least $v \in V$ such that $\langle e_0, ..., e_{n-1}, v\rangle \notin S$. Where we order V by considering $|V| \subset \mathbb{N}$, to obtain our meaning of "least" in this context. Thus, the set $E = \{e_0, e_1, ...,\}$ is a basis for V.
To prove the opposite direction, we reason in RCA_0. We will prove ACA_0 with the proposition that 'every countable vector space over $\mathbb{Q}$ has a basis'. The general principle implies this principle, and proving the $\mathbb{Q}$ case implies ACA_0 will complete the chain of implications and imply the opposite direction. Let $f : \mathbb{N} \to \mathbb{N}$ be a one-to-one function. By Corollary 3.3, it suffices to show the range of this function exists to imply Arithmetic Comprehension. Let V_0 be the set of sums $\sum_{i\in I} q_i \cdot x_i$ where $I \subset N$, I is finite, and $0 \neq q_i \in \mathbb{Q}$. Thus V_0 is a vector space over $\mathbb{Q}$ and $X = \{x_n : n \in \mathbb{N}\}$ is a basis of V_0. For each $m \in \mathbb{N}$ put $x'_m = x_{2f(m)} + m \cdot x_{2f(m)+1}$, and let U be the subspace of V_0 generated by $X' = \{x'_m : m \in \mathbb{N}$. U exists by Δ^0_1 comprehension since $\sum_{i\in I} q_i \cdot x_i$ belongs to U iff $\forall n(q_{2f(n)} \neq 0 \to q_{2f(n)+1}/q_{2f(n)} = n)$ and $\forall n(q_{2n} = 0 \to q_{2n+1} = 0)$. We note that in fact X' is a basis of U. Since U is a subspace of V_0, we may form the quotient space $V = V_0/U$ as follows. The elements of V are those $v \in V_0$ such that $\forall w((w < b \wedge w \in V_0) \to v - w \notin U)$, i.e., v is the minimal representative of an equivalence class under the equivalence relation $v - w \in U$. We define the vector space operations on V accordingly. Thus V is a vector space over $\mathbb{Q}$. By our assumption, V has a basis, call it X''. It follows that $X' \cup X''$ is a basis of V_0. Now we have that for any $n \in \mathbb{N}$, we have $\exists m(f(m) = n)$ if and only ifat least one of the unique expressions for x_{2n} and x_{2n+1} in terms of the basis $X' \cup X''$ involves an element x'_m from X' such that $f(m) = n$. So, by Δ^0_1 comprehension the range of f exists.

□

Acknowledgments

I would like to thank my advisor, Miles Kretschmer, for his unceasing help and patience with the writing of this paper. I would also like to think Peter May for organizing the REU in the first place.

References

[1] Denis R Hirschfeldt. Slicing The Truth. World Scientific Publishing Co. Pte. Ltd. 2015
[2] Robert I. Soare. Turing Computability. Springer-Verlag Berlin Heidelberg. 2016

[3] Stephen G. Simpson. Subsystems of Second Order Arithmetic. Cambridge University Press. 2009

[4] Damir D. Dzhafarov and Carl Mummert. Reverse Mathematics. Springer Nature Switzerland AG. 2022