The Paris-Harrington Theorem

Ville Salo and Ilkka Törmä

[1] University of Turku
vosalo@utu.fi
[2] University of Turku
iatorm@utu.fi

1 Introduction

In Ramsey theory, very large numbers and fast-growing functions are more of a rule than an exception. The classical Ramsey numbers $R(n, m)$ are known to be of exponential size: the original proof directly gives the upper bound $R(n, m) \leq \binom{m+n-2}{n-1}$, and an exponential lower bound is also known. For the van der Waerden numbers, the original proof produced upper bounds that were not even primitive recursive in the case of 2 colors, and even Shelah's improved proof only gives the bound $W(n) = O(g_4(n))$, where $g_4(n) = g_3^n(1)$ and $g_3(1) = 2, g_3(n) = 2^{g_3(n-1)}$.

There are even Ramsey-type functions which provably do not have primitive recursive lower bounds, implying that the Ramsey-type theorems that show their existence cannot be proven in logical systems that can only handle primitive recursion. Peano Arithmetic is a system capable of dealing with primitive recursion and beyond, but even in this system, natural Ramsey theoretic questions can be asked for which a solution exists (in standard set theory), but the existence cannot be proven. The proof of this result is in close connection with ordinal theory, and more precisely the question of how large ordinals a logical system can prove to be well-ordered.

In this essay, following the presentation of [2], we introduce a natural generalization of the classical Ramsey theorem, and prove it using the infinite Ramsey theorem and a compactness argument. We then show, using ordinal theory and a classical result from logic, that the theorem cannot be proved in Peano Arithmetic. Apart from the result from logic, the 'proof of unprovability' is largely self-contained.

2 Structure of the Essay

The structure of our proof follows closely that of [2]. We make a few additional remarks and prove some lemmas that were left as exercises for the reader. We also implement the algorithms in the Python programming language and present some example calculations, in particular, all the examples given in the book.

In Section 3, we present the generalized Ramsey theorem and show how it follows from the infinite Ramsey theorem, and reduce it to the exact question we will actually show unprovable in Peano Arithmetic. In Section 4, we introduce the ordinals and give an effective notation for the subset of them that we need in our proof. Our proof is based on the fact that there exist very large counterexamples to the generalized Ramsey theorem. These are obtained from colorings on ordinals, presented in Section 5, which are then transformed into colorings on natural numbers in Section 6.

In the appendices, we illustrate the theoretical computability of the various functions defined in this essay by implementing them in Python. Python has built-in 'bignums', meaning that there is no a priori limit on the size of natural numbers used. For example, googol (10^{100}) and even 10^{100000} can be computed with a modern computer in a matter of seconds. Still, for instance, for the third function f_3 in an Ackermann-style hierarchy we define, the values $f_3(3)$ and $f_3(4)$ are already way too large: our implementation is too slow for attacking `ack(3, 3)`, of whose magnitude we have little clue, and running `ack(3, 4)` will (quite gracefully) run out of memory.

Example runs of the algorithms performed in the Python read-eval-print-loop IDLE are shown in Appendix A, and the actual (undocumented) program listings are given in Appendix B. See "http://pastebin.com/ve2ncBU0" for a copyable version. The implementation is a rather direct translation from [2] into Python, and agrees with the explicit values given there except for $\omega^{\omega^{\omega}}((3))$ (see Appendix A), which is claimed to have 43 terms in [2], but has only 27 according to our implementation. Note that in 1980 when the book was written, this may still have been computed by hand, leaving a margin of error in any of the 37 reduction steps.

3 The Strengthened Ramsey Theorem

We start by proving a seemingly mild strengthening of the classical Ramsey theorem. For $m \in \mathbb{N}$ (and later for any lattice with a 0, in particular the ordinals), we use the notation $[m]$ for $\{n \in \mathbb{N} \mid 0 \leq n < m\}$ and for a set X, $[X]^k$ for the size k subsets of X. The notation $[[m]]^k$ is shortened to just $[m]^k$.

Definition 1. *Let $h : \mathbb{N} \to \mathbb{N}$ be increasing. A set $X \subset \mathbb{N}$ is h-*large *if $|X| \geq h(\min X)$. For $r, k, n, m \in \mathbb{N}$, we denote $m \xrightarrow{h} (n)^k_r$ if for all r-colorings of $[m]^k$, there exists an h-large monochromatic subset of $[m]$ of size at least n.*

Theorem 1. *Let $h : \mathbb{N} \to \mathbb{N}$ be increasing and $r, k, n \in \mathbb{N}$. Then there exists $m \in \mathbb{N}$ with $m \xrightarrow{h} (n)^k_r$.*

Proof. Let $[\mathbb{N}]^k$ be r-colored. By the infinite Ramsey theorem, we have an infinite monochromatic set $X \subset \mathbb{N}$, and the first $\max(h(\min X), n)$ elements of X form a finite h-large monochromatic set Y of size at least n. By the compactness principle, we have an upper bound for $\min Y$, then for $\min(Y - \{\min Y\})$, and so on. The existence of a finite bound m now follows. $\square$

Theorem 1 is not particularly interesting in itself, since the proof involved only the infinite Ramsey theorem and a standard compactness argument. Its signifigance lies in the following deep result.

Theorem 2 (Paris-Harrington). *Peano arithmetic cannot prove Theorem 1.*

For this statement to be meaningful, we give a quick overview of the system of Peano Arithmetic.

Definition 2. *The system of* Peano Arithmetic *(PA for short) consists of the basic axioms of first-order logic and equality, the constant* 0*, the unary function symbol S (successor), the binary function symbols $+$ and $\cdot$ (addition and multiplication), and the following axioms.*

1. $\forall x : 0 \neq S(x)$
2. $\forall x, y : S(x) = S(y) \implies x = y$
3. $\forall x : x + 0 = x$
4. $\forall x, y : x + S(y) = S(x + y)$
5. $\forall x : x \cdot 0 = 0$
6. $\forall x, y : x \cdot S(y) = x \cdot y + x$

For each first-order predicate $\phi(x, y_1, \ldots, y_n)$, we also have the induction axiom

7. $\forall \boldsymbol{y} : (\phi(0, \boldsymbol{y}) \wedge \forall x : (\phi(x, \boldsymbol{y}) \implies \phi(S(x), \boldsymbol{y})) \implies \forall x : \phi(x, \boldsymbol{y}))$,

where $\boldsymbol{y}$ denotes the vector $(y_1, \ldots, y_n)$.

It is not at all trivial how one can express the statement of Theorem 1 in the language of PA. We merely note that most properties of finite combinatorial structures can be represented as arithmetical statements about natural numbers.

Definition 3. *Define $e_1(n) = n$ and $e_{s+1}(n) = n^{e_s(n)}$, and denote $h_s(n) = e_{s-1}(n) + s - 1$. It can be proven in PA that all h_s are total functions. Define also $P(s, m)$ as the predicate $m \xrightarrow{h_{s+1}} (s)^{s+2}_{2s+1}$.*

In the following, we prove that PA cannot prove the proposition

$$\forall s : \exists m : P(s, m),$$

which implies Theorem 2.

4 Ordinals

The natural numbers $\mathbb{N}$ are defined by $0 \in \mathbb{N}$, and $S(n) \in \mathbb{N}$ for all $n \in \mathbb{N}$, for an injective successor function S. The *ordinals* **Ord** are a sequential compactification of this set, in the sense that every sequence must have a limit point in the interval topology, which remains well-ordered. Thus, the properties we want ordinals to have are

- $0 \in$ **Ord**.
- Whenever $\alpha \in$ **Ord**, we have $S(\alpha) \in$ **Ord**.
- Whenever $\alpha_i \in$ **Ord** is sequence of ordinals indexed by the natural numbers, there exists $\alpha \in$ **Ord** such that $\alpha_{i_j} \to \alpha$ for some subsequence i_j.

– The class **Ord** is well-ordered.

Ordinals have a rich theory in the field of set theory, where they can be defined as, for example, equivalence classes of well-orderings.

The class of all ordinals is unnecessarily large for our purposes (in particular, not all ordinals are accessible using the limit operation described above). However, up to the countable ordinal ε_0, we have the following concrete representation.

Definition 4. *We define the set* **HOrd** *of* harmless ordinals *as follows. First, every $n \in \mathbb{N}$ is a harmless ordinal. Then, if $n_i \in \mathbb{N} - \{0\}$ and $\alpha_i \in$* **HOrd** *for all $i = 1, \ldots, t$ with $\alpha_i > \alpha_{i+1}$, then*

$$\alpha = n_1 \omega^{\alpha_1} + \cdots + n_t \omega^{\alpha_t}, \tag{1}$$

is a harmless ordinal. The order of harmless ordinals with the same exponents is just the lexicographic order. We say α is a limit ordinal *if $\alpha_t > 0$. We also define $\gamma_0 = 1$ and $\gamma_{n+1} = \omega^{\gamma_n}$, and $\varepsilon_0 = \lim_n \gamma_n$.*

Note that each γ_n is harmless, but ε_0 is not. In fact, it is the smallest non-harmless ordinal. We take it for granted that the set of ordinals less than ε_0 is precisely **HOrd**, and in the usual set-theoretic model of ordinals, we actually have $\varepsilon_0 = \mathbf{HOrd}$. The above representation is analogous to the base-b representation of a natural number, since for every pair of numbers $n, b \in \mathbb{N}$, n has a unique base-b representation

$$n = q_1 b^{n_1} + \cdots + q_t b^{n_t}.$$

We could also define arithmetics on harmless (or general) ordinals, so that the formal sum in (1) becomes an actual sum of ordinals, but this is not necessary for our purposes.

If α is not a limit ordinal, it is an isolated point in the interval topology of ordinals. In the following, we show the converse, which justifies the term limit ordinal.

Definition 5. *Let $\alpha \in$* **HOrd** *with the representation* (1)*. The* natural sequence *of α is the sequence $(\alpha(n))_{n \in \mathbb{N}}$, defined as follows. If α is not a limit ordinal, then $\alpha(n) = \alpha$ for all n. Otherwise,*

$$\alpha(n) = n_1 \omega^{\alpha_1} + \cdots + (n_t - 1)\omega^{\alpha_t} + T, \tag{2}$$

where $T = n\omega^{\beta}$, if $\alpha_t = \beta + 1$, and $T = \omega^{\alpha_t(n)}$, if α_t is a limit ordinal. Note that $\alpha_t < \alpha$, so the natural sequence of α_t is defined by the induction hypothesis.

Lemma 1. *With α as above, we have $\lim_n \alpha(n) \to \alpha$.*

Proof. The claim is trivial, if α is not a limit ordinal. Otherwise, it is clear that $\alpha(n) < \alpha(n+1) < \alpha$ for all $n \in \mathbb{N}$. Suppose that $\alpha > \alpha' > \alpha(n)$ for some $\alpha' \in \mathbf{HOrd}$ and all $n \in \mathbb{N}$. Then necessarily

$$\alpha' = n_1 \omega^{\alpha_1} + \cdots + (n_t - 1)\omega^{\alpha_t} + n_{t+1}\omega^{\beta} + T,$$

where $T < \omega^\beta$ and $\beta < \alpha_t$. If α_t is not a limit ordinal, then $\alpha(n_{t+1}+1) > \alpha'$, a contradiction. If α_t is a limit ordinal, then by transfinite induction we have $\alpha_t(n) \to \alpha_t$, in particular $\alpha_t(n) \in (\beta, \alpha_t)$ when n is large enough. This is again a contradiction. $\square$

Definition 6. *Let $\alpha \in$ **HOrd** with the representation* (1)*. We define $T(\alpha) = t$, and*

$$N(\alpha) = \max\{n_1, \dots, n_t, N(\alpha_1), \dots, N(\alpha_t)\} + 1.$$

The quantity $N(\alpha)$ is again defined by transfinite induction, since $\alpha_i < \alpha$.

From the definition of $\alpha(n)$, we see that $N(\alpha(n)) \leq \max(N(\alpha), n)$. The functions T and N have the following relation, which will be used in Section 5.

Lemma 2. *Let $s > 0$. If $\alpha < \gamma_{s+1}$, then $T(\alpha) \leq e_s(N(\alpha))$.*

Proof. We prove, by induction on s, that the number of ordinals $\alpha < \gamma_s$ with $N(\alpha) \leq n$ is $e_s(n)$, from which the claim follows. For $s = 1$, the claim is trivial. Let then $s > 1$, and let $\alpha_1, \dots, \alpha_t$ be the set of ordinals $\beta < \gamma_{s-1}$ with $N(\beta) \leq n$. Then if $\alpha < \gamma_s$ and $N(\alpha) \leq n$, it is of the form $n_1\omega^{\alpha_1} + \cdots + n_t\omega^{\alpha_t}$ with each $n_i \in [0, n-1]$. The number of such α is exactly $n^t = n^{e_{s-1}(n)} = e_s(n)$. $\square$

Lemma 3. *Let $\alpha \in$ **HOrd** be a limit ordinal and $\beta < \alpha$. If $n \geq N(\beta)$, then $\alpha(n) > \beta$.*

Proof. Suppose without loss of generality that $\alpha(1) \leq \beta$. Let α have the representation (1), so that β is of the form $n_1\omega^{\alpha_1} + \cdots + (n_t - 1)\omega^{\alpha_t} + R$ for some term $R < \omega^{\alpha_t}$. But the term T in (2) is greater than R, when $n \geq N(\beta)$, which is seen directly if α_t is not a limit ordinal, and otherwise by the induction hypothesis. $\square$

One reason for the definition of these ordinal sequences is the following construction of very quickly growing functions on the natural numbers. The function $(\alpha, n) \mapsto f_\alpha(n)$ is called the Ackermann hierarchy in [2]. There are many definitions for the Ackermann hierarchy, and the one we use explodes into astronomical values (and beyond) particularly fast.

Definition 7. *Let $\alpha \in$ **HOrd**. We define the (increasing) function $f_\alpha : \mathbb{N} \to \mathbb{N}$ as follows:*

- $f_1(n) = 2n$,
- $f_\alpha(n) = f_\beta^n(n)$, *if $\alpha = \beta + 1$, and*
- $f_\alpha(n) = f_{\alpha(n)}(n)$, *if α is a limit ordinal.*

We also define $f_{\varepsilon_0}(n) = f_{\gamma(n)}(n)$.

Lemma 4. *The mapping $\alpha \mapsto f_\alpha$ is an order-embedding of* **HOrd** $\cup \{\varepsilon_0\}$ *to $(\mathbb{N}^\mathbb{N}, \leq)$, where $f \leq g$ holds if $f(n) \leq g(n)$ for large enough $n \in \mathbb{N}$.*

Proof. Let $\beta < \alpha \leq \varepsilon_0$, and let $n \geq N(\beta)$. We prove by transfinite induction on α that $f_\beta(n) \leq f_\alpha(n)$. If α is not a limit ordinal, let $\alpha = \delta+1$. Now $f_\alpha(n) = f_\delta^n(n) > f_\delta(n)$, since the functions are increasing, and by the induction hypothesis $f_\delta(n) \geq f_\beta(n)$. Suppose then that α is a limit ordinal, so that $\beta < \alpha(n)$ by Lemma 3. Then $f_\alpha(m) = f_{\alpha(m)}(m) > f_\beta(m)$ holds for all $m \geq n$ by the induction hypothesis, and the claim is proved. ◻

In particular, we have that $f_{\varepsilon_0} > f_\alpha$ for all $\alpha \in \mathbf{HOrd}$. The function f_{ε_0} grows unimaginably fast, and in fact, the growth rate is too large for PA to handle in the following sense.

Theorem 3 ([1]). *Let $\phi(n,k)$ be a binary predicate such that $\forall n : \exists k : \phi(n,k)$ is provable in PA. Let $f_\phi(n)$ be the smallest k for which $\phi(n,k)$ holds. Then there exists $\alpha \in \mathbf{HOrd}$ with $f_\phi < f_\alpha$.*

In particular, PA cannot prove that f_{ε_0} is a total function. Similarly, the unprovability of Theorem 1 follows if we can show that $f_P > f_\alpha$ for any harmless α.

5 Colorings of Ordinals

In this section, we define colorings of subsets of $\mathbf{HOrd}$, which can then be translated into colorings of natural numbers. We begin by defining another sequence for all harmless ordinals, and some auxiliary functions.

Definition 8. *Let $\alpha, \delta \in \mathbf{HOrd}$ with α having the representation (1). Then $v_\delta(\alpha) = n_i$, if $\delta = \alpha_i$, and 0, if $\delta \notin \{\alpha_1, \ldots, \alpha_t\}$. If $\beta < \alpha$, we define $\overline{\alpha\beta} = \max\{\delta \mid v_\delta(\alpha) \neq v_\delta(\beta)\}$.*

The function $\overline{\alpha\beta}$ gives the 'position' in the representations of α and β where the lexicographic order 'decides' that $\alpha > \beta$, and in particular, $v_{\overline{\alpha\beta}}(\alpha) > v_{\overline{\alpha\beta}}(\beta)$.

We now define a base coloring on $\mathbf{HOrd}$, from which we later construct more complicated colorings.

Definition 9. *The notation $\{\alpha_1, \ldots, \alpha_r\}_>$ stands for the set $\{\alpha_1, \ldots, \alpha_r\}$, with the added relation $\alpha_1 > \cdots > \alpha_r$.*

Definition 10. *We define the coloring $\chi^* : [\mathbf{HOrd}]^3 \to \{0,1,2\}$ by*

$$\chi^*(\{\alpha,\beta,\gamma\}_>) = \begin{cases} 0, \textit{ if } \overline{\alpha\beta} > \overline{\beta\gamma}, \\ 1, \textit{ if } \overline{\alpha\beta} = \overline{\beta\gamma}, \\ 2, \textit{ if } \overline{\alpha\beta} < \overline{\beta\gamma}. \end{cases}$$

The coloring χ^* has the following useful property.

Lemma 5. *Let $S = \{\alpha_1, \ldots, \alpha_r\}_> \subset \mathbf{HOrd}$ be χ^*-monochromatic. The size of S is bounded depending on the color as follows.*

1. *If* $\chi^*(S) = 0$ *and* $\alpha_1 < \omega^\omega$, *then* $r \le N(\alpha_1) + 1$.
2. *If* $\chi^*(S) = 1$, *then* $r \le N(\alpha_1)$.
3. *If* $\chi^*(S) = 2$, *then* $r \le T(\alpha_1) + 1$.

Proof. Let $\beta_i = \overline{\alpha_i \alpha_{i+1}}$ for all $i \in [1, r-1]$.

1. Let $\omega^s \le \alpha_1 < \omega^{s+1}$ for some $s \in \mathbb{N}$. Now $s \ge \beta_1 > \cdots > \beta_{r-1} \ge 0$, so $r \le s + 2 \le N(\alpha_1) + 1$.
2. Now the β_i are all equal, and we denote them by δ. By definition we have $v_\delta(\alpha_1) > \cdots > v_\delta(\alpha_r)$, so $v_\delta(\alpha_1) \ge r - 1$, and thus $N(\alpha_1) \ge r$.
3. Now we have $\beta_1 < \cdots < \beta_{r-1}$, and it follows that $v_{\beta_i}(\alpha_1) \ne 0$ for all i. Thus $r \le T(\alpha_1) + 1$. □

We now define, for each $s \in \mathbb{N}$, a $(2s-1)$-coloring χ_s of the $(s+1)$-subsets of $[\gamma_s]$ such that if $S \subset [\gamma_s]$ is monochromatic for χ_s, then $|S| \le h_s(N(\max S))$. Recall here that $h_s(n) = e_{s-1}(n) + s - 1$. To begin with, we take $\chi_2 = \chi^*|_{[\gamma_2]}$. Note that the extra assumption on 0-colored sets in Lemma 5 holds, and that $h_2(n) = n+1$. In the case $\chi^*(S) = 2$, Lemma 2 implies that $|S| \le e_1(N(\max S)) + 1 = h_2(N(\max S))$.

Now that the base case is dealt with, we proceed to the induction step. Let $s > 2$, and suppose that χ_{s-1} has been defined and has the property stated above. Let $T = \{\alpha_1, \dots, \alpha_{s+1}\}_> \subset [\gamma_s]$. If $\chi^*(\{\alpha_1, \alpha_2, \alpha_3\}) = 1$ or 2, we set $\chi_s(T) = 2s - 3$ or $2s - 2$, respectively. Otherwise, let $\alpha_i' = \overline{\alpha_i \alpha_{i+1}}$. Note that $\alpha_i' < \gamma_{s-1}$, so that χ_{s-1} is defined on the set $R = \{\alpha_1', \dots, \alpha_s'\}$. In the case that $\alpha_1' > \cdots > \alpha_s'$, we set $\chi_s(T) = \chi_{s-1}(R)$, and otherwise we set $\chi_s(T) = 0$.

Now that χ_s is defined, let $S = \{\alpha_1, \dots, \alpha_r\}_> \subset [\gamma_s]$ be monochromatic for it. We need to show that $r \le h_s(N(\alpha_1))$. First, if $\chi_s(S) = 2s - 3$ or $2s - 2$, then the set $\{\alpha_1, \dots, \alpha_{r-s+2}\}$ is monochromatic for χ^* with color 1 or 2, respectively. Lemmas 5 and 2 then imply that $r - s + 2 \le N(\alpha_1)$ or $r - s + 2 \le e_{s-1}(N(\alpha_1)) + 1$, respectively, and $r \le h_s(N(\alpha_1))$ follows by the definition of h_s.

Suppose then that the χ_s-color of S is not $2s - 3$ or $2s - 2$, and denote $\alpha_i' = \overline{\alpha_i \alpha_{i+1}}$ for all $i \in [1, r-1]$. By the definition of χ_s, we necessarily have $\chi^*(\{\alpha_i, \alpha_{i+1}, \alpha_{i+2}\}) = 0$ when $i \le r - s$, so that $\alpha_i' > \alpha_{i+1}'$ by the definition of χ^*. In other words, $\alpha_1' > \cdots > \alpha_{r-s+1}'$ holds.

Let $1 \le i_1 < \cdots < i_{s-1} \le r - s + 1$ be an arbitrary sequence of indices. Then we have $\alpha_{i_k}' = \overline{\alpha_{i_k} \alpha_{i_{k+1}}}$ for all k, which is easily seen by the definition of $\overline{\alpha\beta}$ (which we recall to be the largest 'coordinate' in which α and β differ). By the definition of χ_s, we then have

$$\chi_{s-1}(\{\alpha_{i_1}', \dots, \alpha_{i_{s-1}}'\}_>) = \chi_s(\{\alpha_{i_1}, \dots, \alpha_{i_{s-1}}, \alpha_{i_{s-1}+1}\}),$$

and thus $\{\alpha_1', \dots, \alpha_{r-s+1}'\}$ is monochromatic under χ_{s-1}. By the induction hypothesis we have $r - s + 1 \le h_{s-1}(N(\alpha_1')) \le h_{s-1}(N(\alpha_1))$, which implies that $r \le h_s(N(\alpha_1))$.

Let us restate the main result of this section as a lemma.

Lemma 6. *For all* $s \in \mathbb{N}$, *there exists a coloring* $\chi_s : [\gamma_s]^{s+1} \to [0, 2s-2]$ *such that if* $S \subset [\gamma_s]$ *is monochromatic for* χ_s, *then* $|S| \le h_s(N(\max S))$.

6 Coloring Natural Numbers and Unprovability in PA

In this section, we use the colorings χ_s to construct colorings of $\mathbb{N}$ that will be used as counterexamples to the predicate P. First, we define new approximating sequences for ordinals as follows.

Definition 11. *Let α be a harmless ordinal. Define the sequence $(\alpha((n)))_{n\in\mathbb{N}}$ by*

$$\alpha((n)) = \lim(\alpha, \alpha(n), \alpha(n)(n), \ldots).$$

The above limit exists and is not a limit ordinal, since we have a decreasing sequence of ordinals, which must be eventually constant.

Remark 1. Let α be a harmless limit ordinal and $n \in \mathbb{N}$. Then there exists an $m \in \mathbb{N}$ with $\alpha((m)) > \alpha(n)$. In particular, we have $\lim_n \alpha((n)) = \alpha$.

Proof. Let $m = N(\alpha(n))$. Now, Lemma 3 implies that $\alpha(m) > \alpha(n)$, $\alpha(m)(m) > \alpha(n)$ and so on, in particular $\alpha((m)) > \alpha(n)$. The latter claim follows, since $\alpha((n)) \leq \alpha((n+1)) \leq \alpha$ for all $n \in \mathbb{N}$. $\square$

These sequences are used to construct certain mappings from **HOrd** to $\mathbb{N}$.

Definition 12. *Let $n \in \mathbb{N}$ and $\alpha \in$ **HOrd**. We define the* ordinal translation function *$T_{n,\alpha} : [n,\infty) \to$ **HOrd** as follows. First, $T_{n,\alpha}(n) = \alpha$. Let then $m \geq n$, and denote $\beta = T_{n,\alpha}(m)$. If $\beta = 0$, then $T_{n,\alpha}(m+1) = 0$, and in all other cases we set $T_{n,\alpha}(m+1) = \beta((m)) - 1$, which is well-defined, since $\beta((m))$ is neither a limit ordinal nor 0. Since $T_{n,\alpha}$ defines a decreasing sequence of ordinals, the case $T_{n,\alpha}(m) = 0$ is eventually reached, and we define $U(n,\alpha)$ to be the least such m.*

Lemma 7. *For all $\alpha \in$ **HOrd** and $n \in \mathbb{N}$, we have $U(n, \omega^\alpha) = f_\alpha(n)$.*

Proof. First, we trivially have $U(n,\omega) = 2n$. Suppose then that the claim holds for all $\beta < \alpha$, and let α be a limit ordinal. Then we have $\omega^\alpha((n)) = \omega^{\alpha(n)}((n))$ by definition, so $T_{n,\omega^\alpha}(n+1) = T_{n,\omega^{\alpha(n)}}(n+1)$ holds. Any value of the translation function determines the rest, and thus

$$U(n,\omega^\alpha) = U(n,\omega^{\alpha(n)}) = f_{\alpha(n)}(n) = f_\alpha(n)$$

by the induction hypothesis.

Suppose then that α is not a limit ordinal, so $\alpha = \beta + 1$ for some $\beta \in$ **HOrd**. We prove, by induction on s, that $U(n, s\omega^\beta) = f^s_\beta(n)$. By the induction hypothesis of β, we have $U(n,\omega^\beta) = f_\beta(n)$, which proves the base case $s = 1$. Let then $s > 1$. Since the operation $(\delta, m) \mapsto \delta(m)$ only changes the last 'digit' of δ, we have $T_{n,s\omega^\beta}(i) = (s-1)\omega^\beta + T_{n,\omega^\beta}(i)$ for all $i \in [n, U(n,\omega^\beta)]$. This implies

$$U(n, s\omega^\beta) = U(U(n,\omega^\beta), (s-1)\omega^\beta) = f^{s-1}_\beta(f_\beta(n)) = f^s_\beta(n)$$

by the induction hypothesis of s. Thus

$$U(n,\omega^\alpha) = U(n, n\omega^\beta) = f^n_\beta(n) = f_\alpha(n),$$

finishing the proof. $\square$

The following is clear from the definition of natural sequences.

Lemma 8. *For all $n, m \in \mathbb{N}$ and $\alpha \in \mathbf{HOrd}$, we have*

$$T_{n,\alpha}(m) \leq \max(N(\alpha), m).$$

In particular, $T(n, \gamma_s)(m) \leq m$.

Using the translation functions, we can now prove the following lemma, which immediately implies Theorem 2.

Lemma 9. *For the predicate P, we have $f_P \geq f_{\varepsilon_0}$.*

Proof. Since $f_{\varepsilon_0}(s) = f_{\gamma_s}(s)$ for all $s \in \mathbb{N}$, it is enough to show that $P(s, n)$ is false whenever $n \leq f_{\gamma_s}(s)$. That is, we simply need to find a counterexample to $P(s, f_{\gamma_s}(s))$, that is, $f_{\gamma_s}(s) \xrightarrow{h_{s+1}} (s)^{s+2}_{2s+1}$.

The sets of $[f_{\gamma_s}(s)]^{s+2}$ containing a number smaller than s can be colored arbitrarily, since they cannot appear in any h_{s+1}-large set of size s. So let us show that we can color the $(s+2)$-subsets of $[s, f_{\gamma_s}(s)]$ with $2s+1$ colors so that there is no monochromatic subset X of size at least $h_{s+1}(\min X)$. Let $T = T_{s,\gamma_{s+1}(s)} = T_{s,\gamma_{s+1}}$ and note that $U(s, \gamma_{s+1}) = f_\gamma(s)(s)$, so that the values $T(j)$ are distinct and decreasing for $j \in [s, f_{\gamma_s}(s)]$.

To obtain the coloring, we transform $\chi_{s+1} : [\gamma_{s+1}]^{s+2} \to 0, \ldots, 2s$ into a coloring $\chi'_{s+1} : [s, f_{\gamma_s}(s)]$ by simply letting

$$\chi'_{s+1}(\{n_1, \ldots, n_{s+2}\}_<) = \chi_{s+1}(\{T(n_1), \ldots, T(n_{s+2})\}_>).$$

Now let X be any h_{s+1}-large monochromatic set for χ'_{s+1}. Then $|X| = |T(X)| \leq h_s(N(T(\min X))) \leq h_s(\min X)$ by Lemma 6 and Lemma 8, since $T(X)$ is monochromatic for χ_{s+1}. □

References

1. G Gentzen. Die Widerspruchfreiheit der reinen Zahlentheorie. Mathematische Annalen 112 (1936) 495565 English translation. In *The Collected Papers of Gerhard Gentzen, North-Holland, 1969. 11 Schtte, K.: Beweistheoretische Erfassung der unendliche Induction in der Zahlentheorie. Mathematische Annalen 122*, pages 369–389, 1950.
2. Ronald L. Graham, Bruce L. Rothschild, and JoelH. Spencer. *Ramsey theory.* Wiley-Interscience Series in Discrete Mathematics. John Wiley & Sons Inc., New York, 1980. A Wiley-Interscience Publication.

Appendix A: Example Computations

We try out all the examples in [2] and the ones mentioned in Section 2, also illustrating the ordinal syntax.
Examples on page 152.

```
>>> fund("w^(2)",101)
'101w'
>>> fund("w^(2)+3w",101)
'w^(2) + 2w + 101'
>>> fund("5w^(w^(2)+2w)",101)
'4w^(w^(2) + 2w) + w^(w^(2) + w + 101)'
>>>
>>> deepfund("w^(2)",5)
'4w + 5'
>>> numterms(deepfund("w^(w^(w))",3))
27
>>> deepfund("w^(w^(w))",3)
'2w^(2w^(2) + 2w + 2) + 2w^(2w^(2) + 2w + 1) + 2w^(2w^(2) + 2w) +
2w^(2w^(2) + w + 2) + 2w^(2w^(2) + w + 1) + 2w^(2w^(2) + w) +
2w^(2w^(2) + 2) + 2w^(2w^(2) + 1) + 2w^(2w^(2)) + 2w^(w^(2) +
2w + 2) + 2w^(w^(2) + 2w + 1) + 2w^(w^(2) + 2w) + 2w^(w^(2) +
w + 2) +2w^(w^(2) + w + 1) + 2w^(w^(2) + w) + 2w^(w^(2) + 2) +
2w^(w^(2) + 1) + 2w^(w^(2)) + 2w^(2w + 2) + 2w^(2w + 1) +
2w^(2w) + 2w^(w + 2) + 2w^(w + 1) + 2w^(w) + 2w^(2) + 2w + 3'
```

Examples on page 152.

```
>>> trans(5, "w^(2)",5)
'w^(2)'
>>> trans(5, "w^(2)",6)
'4w+4'
>>> trans(5, "w^(2)",10)
'4w'
>>> trans(5, "w^(2)",11)
'3w+9'
>>> trans(5, "w^(2)",20)
'3w'
>>> trans(5, "w^(2)",40)
'2w'
>>> trans(5, "w^(2)",80)
'w'
>>> trans(5, "w^(2)",160)
'0'
>>> u(5, "w^(2)")
'160'
```

Computing $f_3(n)$ for various n. Or at least trying.

```
>>> ack("3",1)
'2'
>>> ack("3",2)
'2048'
>>> ack("3",3)
'Sorry, I ran out of time.'
>>> ack("3",4)
'I need a bigger universe to compute this.'
```

In fact, our implementation cannot really compute $f_\alpha(n)$ for any proper ordinal α if $n > 2$.

Appendix B: Implementation in Python

The program listing can be found at "http://pastebin.com/ve2ncBU0", and is written for Python 2.5.1.

```
import time
timeout_in_sec = 3

"""
The functions numterms, add, deepfund, fund, ack, trans, u and
beautify take their natural number inputs as normal Python
integers, and their ordinal inputs in the format
<ord> = <ord> + <ord>
      | <num>w^(<ord>)
      | <num>w
      | <num>
with the usual ordinal addition rules. The notation only
covers the harmless ordinals.
"""

def numterms(alpha):
    """
    Returns the number of terms in alpha.
    """
    return len(p(alpha))

def add(*alphas):
    """
    Calculates alpha + beta with the usual non-commutative
    ordinal addition.
    """
    return up(p("+".join(alphas)))

def deepfund(alpha, n):
    """
    Calculates alpha((n)) given an ordinal alpha and a
    natural number n.
    """
    return up(timeout(deepfund_, (p(alpha),n),
                      td=timeout_in_sec))

def fund(alpha, n):
    """
    Calculates alpha(n) given an ordinal alpha and a natural
    number n.
    """
```

```
    return up(timeout(fund_, (p(alpha),n), td=timeout_in_sec))

def ack(alpha, n):
    """
    Calculates f_alpha given an ordinal alpha.
    """
    return up(timeout(timeout(ack_, (p(alpha),),
                              td=timeout_in_sec),
                      (n,),
                      td=timeout_in_sec))

def trans(n, alpha, m):
    """
    Calculates T_{n, alpha} as a generator, given an ordinal
    alpha and a natural number n.
    """
    t = timeout(trans_, (n, p(alpha)), td=timeout_in_sec)
    v = 0
    for i in range(m-n+1):
        v = t.next()
    return up(v)

def u(n, alpha):
    """
    Calculates U(n, alpha) given an ordinal alpha and a natural
    number n.
    """
    return up(timeout(u_, (n, p(alpha)), td=timeout_in_sec))

def beautify(o):
    """
    Fixes the representation of an ordinal.
    """
    return up(p(o)) # nalle

class ParseError(Exception):
    def __init__(self, value):
        self.value = value
    def __str__(self):
        return repr(self.value)

def p(o):
    if isinstance(o, int): return o
    else:
        o = "".join(filter(lambda a:a!=" ", o))
```

```
        try:
            parsed = p_(o,0)[0]
        except ParseError, e:
            return "Parse error near " + str(e.value) + \
                   " in ordinal input."
        except:
            return "Unknown parse error in ordinal input."
    return fix_(parsed)

def up(o):
    if isinstance(o,str):return o
    o = fix_(o)
    if isinstance(o, int):return str(o)
    def m(t):
        if isinstance(t, int):return str(t)
        elif t[1]==1:
            r="w"
        else:
            r="w^("+up(t[1])+")"
        if t[0]==1:return r
        return str(t[0])+r
    return " + ".join(map(m,o))

def p_(string,i):
    v = []
    while True:
        j=i
        while j < len(string) and string[j].isdigit():
            j+=1
        intstring = string[i:j]
        if intstring == "":
            coef = None
        else:
            coef = int(intstring)
        i=j
        if i == len(string) or string[i] == ")":
            if coef == None: return v, i+1
            v.append(coef)
            return v, i+1
        elif string[i:i+3] == "w^(":
            exp, i = p_(string, i+3)
            if coef == None: coef = 1
            v.append((coef, exp))
        elif string[i]=="w":
            i+=1
```

```
            if coef == None: coef = 1
            v.append((coef, 1))
        elif string[i]=="+":
            i+=1
        else:
            raise ParseError(i)
        if i < len(string) and string[i] == "+":
            i+=1

def timeout(func, args=(), td=None,
            default="Sorry, I ran out of time."):
    import threading
    class InterruptableThread(threading.Thread):
        def __init__(self):
            threading.Thread.__init__(self)
            self.result = default
        def run(self):
            try:
                self.result = func(*args)
            except:
                self.result = punchline
    it = InterruptableThread()
    it.start()
    if td == None:
        it.join()
    else:
        it.join(td)
    return it.result

def limit_(alpha):
    if isinstance(alpha, int):
        return False
    lastv = alpha[-1][1]
    return lastv != 0

def ord_cmp_(alpha, beta):
    alpha, beta = fix_(alpha), fix_(beta)
    # base cases
    if isinstance(alpha, int):
        if isinstance(beta, int):
            return alpha - beta
        return -1
    if isinstance(beta, int):
        return 1
    c = ord_cmp_(alpha[0][1], beta[0][1])
```

```
    if c == 0:
        coefc = alpha[0][0] - beta[0][0]
        if coefc == 0:
            return ord_cmp_(alpha[1:], beta[1:])
        return coefc
    return c

def fix_(alpha):
    if isinstance(alpha, int):
        return alpha
    if len(alpha) == 0:
        return 0
    def intfix_(a):
        if isinstance(a, int):return (a,0)
        return a
    def unintfix_(a):
        if isinstance(a, int):a
        elif a[1]==0:return a[0]
        return a
    alpha = list(map(intfix_, alpha))
    if len(alpha) == 1 and fix_(alpha[0][1]) == 0:
        return alpha[0][0]
    alpha = list(map(lambda(c,e):(c,fix_(e)),alpha))
    i = 1
    while i < len(alpha):
        if i == 0:
            i+=1
            continue
        c = ord_cmp_(alpha[i-1][1], alpha[i][1])
        if c < 0:
            alpha = alpha[0:i-1] + \
                    [(alpha[i][0], alpha[i][1])] + \
                    alpha[i+1:]
            i -= 1
        elif c == 0:
            alpha = alpha[0:i-1] + \
                    [(alpha[i-1][0] + alpha[i][0],
                      alpha[i][1])] + alpha[i+1:]
        elif c > 0:
            i += 1
    return list(map(unintfix_,alpha))

def prec_(alpha):
    assert not limit_(alpha) and alpha != 0 and \
           alpha != [(0,0)]
```

```
    if isinstance(alpha, int):
        return alpha - 1
    beta = alpha[:]
    if beta[-1][0] == 1:
        return fix_(beta[:-1])
    beta[-1] = (beta[-1][0]-1,beta[-1][1])
    return beta

def fund_(alpha,n):
    if not limit_(alpha):
        return alpha
    beta = alpha[:-1]
    lastc, lastv = alpha[-1]
    if lastc > 1:
        beta = beta + [(lastc-1,lastv)]
    if limit_(lastv):
        return beta + [(1,fund_(lastv,n))]
    else:
        return beta + [(n,prec_(lastv))]

def power_(f,n):
    def pow_f_n(m):
        k=m
        for i in xrange(n):
            k=f(k)
        return k
    return pow_f_n

def deepfund_(alpha,n):
    beta=alpha
    while limit_(beta):
        beta=fund_(beta,n)
    return beta

def ack_(alpha):
    if alpha == 1 or alpha == [(1,0)]:
        return lambda n: n*2
    if not limit_(alpha):
        return lambda n: power_(ack_(prec_(alpha)),n)(n)
    return lambda n: ack_(fund_(alpha,n))(n)

def trans_(n,alpha):
    beta = alpha
    m=n
    while True:
```

```
        yield beta
        if beta!=0 and beta!=[(0,0)]:
            beta = prec_(deepfund_(beta,m))
        m+=1

def u_(n,alpha):
    g=trans_(n,alpha)
    m=n
    while True:
        beta=g.next()
        if beta==0 or beta==[(0,0)]:
            return m
        m+=1

punchline = "I need a bigger universe to compute this."
```