The Hydra Battle Revisited*

Nachum Dershowitz[1,2] and Georg Moser[3]

[1] School of Computer Science, Tel Aviv University, 69978 Ramat Aviv, Israel,
nachum.dershowitz@cs.tau.ac.il
[2] Microsoft Research, Redmond, WA 98052
[3] Institute of Computer Science, University of Innsbruck, Technikerstrasse 21a,
A-6020 Innsbruck, Austria, georg.moser@uibk.ac.at

To Jean-Pierre on this momentous occasion.

Abstract. Showing termination of the Battle of Hercules and Hydra is a challenge. We present the battle both as a rewrite system and as an arithmetic while program, provide proofs of their termination, and recall why their termination cannot be proved within Peano arithmetic.

As a second labour he ordered him to kill the Lernaean hydra.
That creature, bred in the swamp of Lerna,
used to go forth into the plain
and ravage both the cattle and the country.
Now the hydra had a huge body, with nine heads,
eight mortal, but the middle one immortal. . . .
By pelting it with fiery shafts he forced it to come out,
and in the act of doing so he seized and held it fast.
But the hydra wound itself about one of his feet and clung to him.
Nor could he effect anything by smashing its heads with his club,
for as fast as one head was smashed there grew up two.

– Pausanias, *Description of Greece*, 2.37.4

1 Introduction

The Battle of Hydra and Hercules, as described in the above-quoted myth, and depicted on the Etruscan hydra (water jar) in Fig. 1, inspired Laurie Kirby and Jeff Paris [24] to formulate a process, the termination of which cannot be proved by ordinary induction on the natural numbers. Instead, recourse must be made to induction on the ordinals less than the ordinal number "epsilon naught", in the ordinal hierarchy created by Georg Cantor.

The alternating steps of Hercules and Hydra in the formal battle are quite easy to understand (and are more appealing than the similar but older Goodstein sequence [17], also treated in [24]). The battle itself is described in Sect. 2. Yet

* The first author's research was supported in part by the Israel Science Foundation (grant no. 250/05).

Fig. 1. Caeretan hydra, attributed to Eagle Painter, c. 525 B.C.E. See [19]. (Courtesy of the J. Paul Getty Museum, Villa Collection, Malibu, CA)

the fact that Hercules is always the declared winner, as shown in Sect. 4, is far from obvious. As such, it is arguably the simplest example of a terminating process that is not amenable to argument by means of Peano's famous axioms of arithmetic. Termination is a conceptually clear notion. Thus, it is fair to claim that the Hydra Battle is more intuitive, as an independence result, than employing Ramsey-like theorems, for instance; cf. [30].

In the popular survey on rewriting [10][4] by Jean-Pierre Jouannaud and the first author (and in several later publications, [11, Prob. 23], [7]), a rewrite system for the battle was presented, but it was unfortunately fraught with *lapsus calami*. Nevertheless, proving its termination has been a challenge for contestants in termination competitions [28]. A repaired version was promulgated years later [26].

In the sections that follow, we describe the vicissitudes of this formalization of the battle in rewrite systems (Sect. 5), prove their termination (Sect. 8), and also encode the battle as a **while** program (Sect. 9). This paper concentrates on the interpretation of successive hydræ as decreasing ordinal numbers. But there are alternative, less "highbrow" arguments (in Alan Turing's words [35]); see Martin Gardner's column [15]. Some properties of ordinals and orders are briefly reviewed in Sects. 3, 6 and 7.

For more on the problem and its extensions, see [13]. See also [27,21,34]. We conclude with one such extension.

2 The Hydra Battle

In a landmark paper [24], Kirby and Paris showed that – for their version of the battle – more than ordinary induction on the natural numbers is needed to show that Hercules prevails.

2.1 The Formal Battle

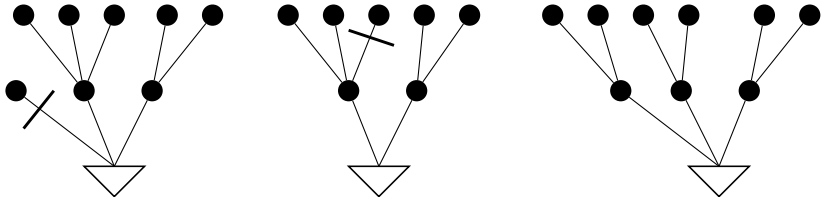
In the mathematical battle, hydræ are represented as (unordered, rooted, finite) trees, with each leaf corresponding to one head of the monster. Whereas Hercules decapitates Hydra, one head at a time, Hydra regenerates according to the following rule: If the severed head has a grandparent node, then the branch issuing from that node together with the mutilated subtree is multiplied by a certain factor; otherwise, Hydra suffers the loss without any regrowth. Hercules wins when (not if!) the beast is reduced to the empty tree.

(In the original formulation of this game, the multiplication factor is one more than the stage of the game. We modify the definition slightly and set the multiplication factor equal to the stage of the game, as otherwise the system $\mathcal{D}$ below fails to encode the Hydra Battle. The results in [24] are unaffected by this change.)

[4] This survey is the most cited document for its year of publication (1990) in the CiteSeer database [31], and has always been high in the overall list.

We write (H, n) to describe a single *configuration* in the game, where H denotes Hydra and n the current stage of the game.

Example 1.



$(\mathsf{H}_1, 1)$ $(\mathsf{H}_2, 2)$ $(\mathsf{H}_3, 3)$

In the first stage, Hercules chops off the leftmost head. As this head has no grandparent, Hydra shrinks. However, in Stage 2, Hercules chops off a head with a grandparent (the triangular root). Consequently, Hydra grows two replacement branches, as indicated. □

More examples can be found in [24,34].

2.2 Functional Hydra

It is easy to express the Hydra Battle in a functional language with list operations, like Lisp. However, so as not to complicate matters unnecessarily, we do not follow the original definition of Kirby and Paris, but rather restrict attention to ordered trees (with immediate subtrees ordered sequentially from left to right). Let `nil` denote an empty list and $\texttt{cons}(x, y)$ be the list obtained by prepending an element x (the right-most subtree) to a list y (the remaining tree). Thus, the first hydra in Example 1 would be represented as follows:

$$\texttt{cons}(\ell, \texttt{cons}(\texttt{cons}(\texttt{cons}(\ell, \texttt{cons}(\ell, \texttt{nil})), \texttt{nil}), \texttt{cons}(\ell, \texttt{cons}(\ell, \texttt{cons}(\ell, \texttt{nil}))))) \ ,$$

where $\ell = \texttt{cons}(\texttt{nil}, \texttt{nil})$ stands for a leaf of a Hydra and we have reordered the immediate subtrees of the root of H_1 such that the number of nodes is non-increasing. In the rewrite system to be introduced below, a "`cons`-cell" is grafted together by the function symbol `g`.

Let $\texttt{car}(\texttt{cons}(x, y)) = x$ and $\texttt{cdr}(\texttt{cons}(x, y)) = y$ extract the first item in a nonempty list $\texttt{cons}(x, y)$ and the remainder of the list, respectively. The battle can be encoded by the following program, $\mathcal{L}$:

$$
\begin{aligned}
&h_0(x)\\
&\text{where}\\
&h_n(x) \quad := \begin{cases} \mathtt{nil} & x = \mathtt{nil} \\ h_{n+1}(d_n(x)) & \text{otherwise} \end{cases}\\
&d_n(x) \quad := \begin{cases} \mathtt{cdr}(x) & \mathtt{car}(x) = \mathtt{nil} \\ f_n(\mathtt{cdr}(\mathtt{car}(x)), \mathtt{cdr}(x)) & \mathtt{car}(\mathtt{car}(x)) = \mathtt{nil} \\ \mathtt{cons}(d_n(\mathtt{car}(x)), \mathtt{cdr}(x)) & \text{otherwise} \end{cases}\\
&f_n(y,x) := \begin{cases} x & n = 0 \\ \mathtt{cons}(y, f_{n-1}(y,x)) & \text{otherwise} \end{cases}
\end{aligned}
$$

Here, $h_n(x)$ plays the game with initial hydra x, starting at stage n, $d_n(x)$ plays one round of the battle, by travelling along a leftmost branch until encountering a branch z such that $\mathtt{car}(\mathtt{car}(z))$ is empty, and then using $f_n(y,x)$ to prepend k copies of $y = \mathtt{cdr}(\mathtt{car}(z))$ to x.

Better yet, if we let ε symbolize `nil` and use a colon : for `cons`, then the following pattern-directed program plays the battle $h_0(x)$ until its inevitable end:

$$
\begin{aligned}
h_n(\varepsilon) &:= \varepsilon\\
h_n(x:y) &:= h_{n+1}(d_n(x:y))\\
d_n(\varepsilon:y) &:= y\\
d_n((\varepsilon:x):y) &:= f_n(x,y)\\
d_n(((u:v):x):y) &:= d_n(((u:v):x):y)\\
f_0(y,x) &:= x\\
f_{n+1}(y,x) &:= y : f_n(y,x)
\end{aligned}
$$

3 Orders and Ordinals

Termination proofs are often based on well-founded orderings. Our proofs are no exception.

3.1 Well-founded Orders

A *partial order* $\succ$ is an irreflexive and transitive binary relation. Its converse is written with a reflected symbol $\prec$. A *quasi-order* $\succcurlyeq$ is a reflexive and transitive relation. A quasi-order $\succcurlyeq$ induces a (strict) partial order $\succ$, such that $a \succ b$ if $a \succcurlyeq b \not\succcurlyeq a$. A partial order $\succ$ on a set A is *well-founded* (on A) if there exists no infinite descending sequence $a_1 \succ a_2 \succ \cdots$ of elements of A. A partial order is *linear* (or *total*) on A if for all $a, b \in A$, a different from b, a and b are comparable by $\succ$. A linear well-founded order is called a *well-order*.

Let $\mathcal{F}$ be a signature. An *$\mathcal{F}$-algebra* $\mathcal{A}$ is a set A, its *domain*, together with operations $f_{\mathcal{A}}\colon A^n \to A$ for each function symbol $f \in \mathcal{F}$ of arity n. An $\mathcal{F}$-algebra $(\mathcal{A}, \prec)$ is called *monotone* if $\mathcal{A}$ is associated with a partial order $\succ$ and every algebra operation $f_{\mathcal{A}}$ is strictly monotone in all its arguments. A monotone $\mathcal{F}$-algebra $(\mathcal{A}, \succ)$ is called *well-founded* if $\succ$ is well-founded.

Let $(\mathcal{A}, \succ)$ denote an $\mathcal{F}$-algebra and let $\mathbf{a}\colon \mathcal{V} \to A$ denote an *assignment*. We write $[\mathbf{a}]_{\mathcal{A}}$ to denote the homeomorphic extension of the assignment $\mathbf{a}$ and define an ordering $\succ_{\mathcal{A}}$ on terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$ in the usual way: $s \succ_{\mathcal{A}} t$ if $[\mathbf{a}]_{\mathcal{A}}(s) \succ [\mathbf{a}]_{\mathcal{A}}(t)$ for every assignment $\mathbf{a}$.

3.2 Order Types

We assume some very basic knowledge of set theory and in particular of ordinals, as in, for example, [22]. We write $>$ to denote the well-order on ordinals. This order can, of course, be employed for inductive arguments.

The ordinal ϵ_0 ("epsilon naught") is the smallest solution to $\omega^x = x$. Recall that any ordinal $\alpha < \epsilon_0$, $\alpha \neq 0$, can be uniquely represented in Cantor Normal Form (CNF) as a sum

$$\omega^{\alpha_1} + \cdots + \omega^{\alpha_n} ,$$

where $\alpha_1 \geqslant \cdots \geqslant \alpha_n$. The set of ordinals below ϵ_0 in CNF will also be denoted CNF. For $\alpha = \omega^{\alpha_1} + \cdots + \omega^{\alpha_n}$ and $\beta = \omega^{\alpha_{n+1}} + \cdots + \omega^{\alpha_{n+m}}$, the *natural sum* $\alpha \oplus \beta$ is defined as $\omega^{\alpha_{\pi(1)}} + \cdots + \omega^{\alpha_{\pi(n+m)}}$, where π denotes a permutation of the indices $[1, n+m]$ $(= \{1, \ldots, n+m\})$ such that $\alpha_{\pi(1)} \geqslant \alpha_{\pi(2)} \geqslant \cdots \geqslant \alpha_{\pi(n+m)}$ is guaranteed. We write $\alpha \cdot n$ as an abbreviation for $\alpha + \cdots + \alpha$ (n times α), and we identify the natural numbers ($\mathbf{N}$) with the ordinals below ω. We denote the set of limit ordinals by Lim.

To each well-founded order $\succ$ on a set A, one can associate a (set-theoretic) ordinal, its *order type*. First, we associate an ordinal to each element a of A by setting

$$\mathrm{otype}_{\succ}(a) := \sup\{\mathrm{otype}_{\succ}(b) + 1 \mid b \in A \text{ and } a \succ b\} .$$

Then the *order type* of $\succ$, denoted $\mathrm{otype}(\succ)$, is defined as $\sup\{\mathrm{otype}_{\succ}(a) + 1 \mid a \in A\}$. For two partial orders $\succ$ and $\succ'$ on A and A', respectively, a mapping $o\colon A \to A'$ *embeds* $\succ$ into $\succ'$ if, for all $x, y \in A$, we have that $x \succ y$ implies $o(x) \succ' o(y)$.

Lemma 1. *If both $\succ$ and $\succ'$ are well founded and if $\succ$ can be embedded into $\succ'$, then* $\mathrm{otype}(\succ) \leqslant \mathrm{otype}(\succ')$.

Two linear partial orders $(A, \succ)$ and $(B, \succ')$ are *order-isomorphic* (or *equivalent*) if there exists a surjective mapping $o\colon A \to B$ such that $(x \succ y \iff o(x) \succ' o(y))$ for all $x, y \in A$.

4 Herculean Strength

The natural game-theoretic question is whether Hercules has a winning strategy. A *strategy* is a mapping determining which head Hercules should chop off at each stage.

4.1 Hercules Prevails

It turns out that whatever strategy Hercules fights by he eventually wins. It's only a question of time. In our proof, we follow Kirby and Paris [24] and associate with each hydra an ordinal below ϵ_0:

1. Assign 0 to each leaf.
2. Assign $\omega^{\alpha_1} \oplus \cdots \oplus \omega^{\alpha_n}$ to each internal node, where α_i are the ordinals assigned to the children of the node.

The ordinal representing Hydra is the ordinal assigned to her root node.

Example 2. Consider hydræ H_1–H_3, above. These have the representations $\omega^3 \oplus \omega^2 \oplus 1$, $\omega^3 \oplus \omega^2$, and $\omega^2 \cdot 3$, respectively. □

In the sequel, we confound the representation of a hydra as finite tree and as ordinal. Let (H, n) denote a configuration of the game. Then $(\mathsf{H})^{\mathsf{S}}_n$ denotes the resulting Hydra if strategy S is applied to H at stage n. That is, the next configuration is $((\mathsf{H})^{\mathsf{S}}_n, n+1)$.

Lemma 2 (Kirby & Paris [24]). *For any strategy* S*, Hydra* H *and natural number* n*, we obtain* $\mathsf{H} > (\mathsf{H})^{\mathsf{S}}_n$.

Theorem 1 (Kirby & Paris [24]). *Every strategy is a winning strategy.*

Proof. The theorem follows from the lemma together with the fact that $>$ is well-founded. □

One can readily see from the proof that Hercules also wins a seemingly more challenging battle, wherein Hydra generates an arbitrary number of replacement branches at any level of the tree, but the resulting Hydra is smaller (as an ordinal) than the original. Again, any strategy is a winning strategy. Such a reformulation of the Hydra Battle is depicted in Fig. 2, taken from a survey lecture on proofs and computation by Jouannaud [23].[5] Similar extensions of the Hydra Battle have recently been considered by Rudolf Fleischer in [13].

Now we formally define a specific strategy for the Hydra Battle that has been called *standard* in [34]. For $n \in \mathbf{N}$, we associate an ordinal $\alpha_n \in \mathrm{CNF}$ with every $\alpha \in \mathrm{CNF}$:

$$\alpha_n = \begin{cases} 0 & \text{if } \alpha = 0 \\ \beta & \text{if } \alpha = \beta + 1 \\ \beta + \omega^\gamma \cdot n & \text{if } \alpha = \beta + \omega^{\gamma+1} \\ \beta + \omega^{\gamma_n} & \text{if } \alpha = \beta + \omega^\gamma \text{ and } \gamma \in \mathsf{Lim} \end{cases}$$

Then we can define the *standard Hydra Battle* as follows:

Definition 1 (Hydra Battle). *A* hydra *is an ordinal in CNF. The* Hydra Battle *is a sequence of configurations. A* configuration *is a pair* (α, n)*, where* α *denotes a hydra and* $n \geqslant 1$*, the current step. Let* (α, n) *be a configuration, such that* $\alpha > 0$*. Then the next configuration in the standard strategy is* $(\alpha_n, n+1)$*.*

[5] "This is one of Nachum Dershowitz's favorite examples" [23].

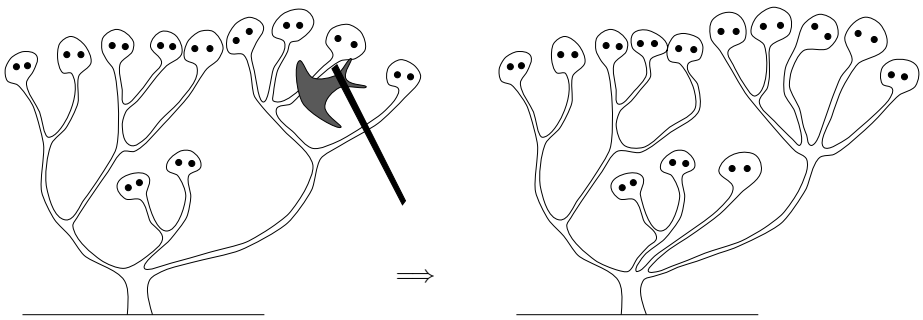


Fig. 2. Evelyne Contejean's rendition of the battle (from Jouannaud's survey [23])

This definition implies that we force a one-to-one correspondence between finite trees and ordinals. Hence, we again restrict ourselves to ordered trees.

This strategy conforms with the prior description of the battle, provided Hydra's immediate subtrees in the functional battle are arranged (at all levels) from the largest on the left to the smallest on the right. Here, largeness or smallness of subtrees is measured by the corresponding ordinal.

Remark 1. The sequence $(\alpha_n)_{n\in\mathbf{N}}$ is usually referred to as a *fundamental sequence* of α. A fundamental sequence fulfills the property that for limit ordinal $\lambda \in \mathsf{Lim}$, the sequence $(\lambda_n)_n$ is strictly increasing and its limit is λ. For the connection between rewriting and fundamental sequences, see, for example, [29].

4.2 Beyond Peano

In the remainder of this section, we show that Peano Arithmetic cannot prove termination of the standard Hydra Battle, which is a special case of a more general theorem stated in [24,13].

We define two ordinal-indexed hierarchies of number-theoretic functions.

Definition 2 (Hardy [36] and Hydra Functions). *The* Hardy functions $(\mathsf{H}_\alpha)_{\alpha<\epsilon_0}$ *are defined as follows:*

$$\mathsf{H}_0(n) := n, \quad \textit{and} \quad \mathsf{H}_\alpha(n) := \mathsf{H}_{\alpha_n}(n+1) \quad (\textit{if } \alpha > 0)\ .$$

The related Hydra functions $(\mathsf{L}_\alpha)_{\alpha<\epsilon_0}$*, counting the length of the (standard) Hydra Battle, starting stage n with hydra α, are:*

$$\mathsf{L}_0(n) := 0, \quad \textit{and} \quad \mathsf{L}_\alpha(n) := \mathsf{L}_{\alpha_n}(n+1)+1 \quad (\textit{if } \alpha > 0)\ .$$

The following lemma is an easy consequence of the definitions:

Lemma 3. *The hierarchies* $(\mathsf{H}_\alpha)_{\alpha<\epsilon_0}$ *and* $(\mathsf{L}_\alpha)_{\alpha<\epsilon_0}$ *form a majorization hierarchy on* ϵ_0*, in the sense that the functions are strictly increasing and each function at level* α *eventually dominates the functions at level* β *for all* $\beta < \alpha$*.*

Next, we relate the functions H_α and L_α.

Lemma 4. *For any* $\alpha < \epsilon_0$ *and any* $n \in \mathbf{N}$ *we have*

$$\mathsf{H}_\alpha(n) = \mathsf{H}_{\mathsf{L}_\alpha(n)}(n) = n + \mathsf{L}_\alpha(n) \ .$$

Proof. The second equation is easily established by noting that, for finite m, we have $\mathsf{H}_m(n) = m + n$. The first equation is proved by induction on α. The case $\alpha = 0$ follows from $\mathsf{L}_0(n) = 0$. For $\alpha > 0$, put $\gamma := \alpha_n$ and note that

$$\mathsf{H}_\alpha(n) = \mathsf{H}_\gamma(n+1) = \mathsf{H}_{\mathsf{L}_\gamma(n+1)}(n+1) = \mathsf{H}_{\mathsf{L}_\gamma(n+1)+1}(n) = \mathsf{H}_{\mathsf{L}_\alpha(n)}(n) \ .$$

In the second equality, we employ the induction hypothesis. For the third let $m = \mathsf{L}_\gamma(n+1)$ and observe $\mathsf{H}_{m+1}(n) = \mathsf{H}_m(n+1)$. □

A function f is provably recursive in Peano Arithmetic (PA) if there exists a primitive recursive predicate P and a primitive recursive function g such that $\mathrm{PA} \vdash \forall y_1 \cdots \forall y_k \exists x P(y_1, \ldots, y_k, x)$ and f satisfies

$$f(n_1, \ldots, n_k) = g(\mu_x P(n_1, \ldots, n_k, x)) \ ,$$

where μ_x denotes the least number operator.

Let the *Hardy class* $\mathcal{H}$ be defined as the smallest class of functions (1) containing 0, S (successor), all H_α, for $\alpha < \epsilon_0$, and all projection functions $I_{n,i}(a_1, \ldots, a_n) := a_i$, and (2) closed under primitive recursion and composition.

Theorem 2. *The Hardy class* $\mathcal{H}$ *is the class of all provably recursive functions in PA.*

For a proof see [32].

Theorem 3. *PA cannot prove termination of the standard Hydra Battle.*

Proof. Suppose termination of the standard Hydra Battle would be PA-provable. This is equivalent to the fact that

$$\mathrm{PA} \vdash \ulcorner \forall\, \alpha, n\ \exists m\ \mathsf{L}_\alpha(n) = m \urcorner \ ,$$

for a suitable arithmetization $\ulcorner \cdot \urcorner$. Hence, for all $\alpha < \epsilon_0$ and all $n \in \mathbf{N}$: $\mathsf{L}_\alpha(n)$ is a provably recursive function in PA. In particular, $\mathsf{L}_{(\epsilon_0)_n}(n+1) + 1$ is provably recursive, and thus by definition $\mathsf{L}_{\epsilon_0}(n)$ is provably recursive. (The same holds for $\mathsf{L}_{\epsilon_0}(n) + n$.) By Lemma 4, we conclude $\mathsf{L}_{\epsilon_0}(n) + n = \mathsf{H}_{\epsilon_0}(n)$, which would imply that $\mathsf{H}_{\epsilon_0}(n)$ is provably recursive, contradicting Theorem 2. □

Remark 2. It bears mentioning that if Hydra is constrained to a bounded initial height, or if there is a fixed bound on her growth factor [27], then termination is provable in PA.

5 Rewriting Hydra

A *(term-) rewriting system* is a (finite) set of rewrite rules, each of which is an ordered pair of terms. Let $\mathcal{F}$ denote a signature and $\mathcal{V}$, a (countably infinite) set of variables. The terms over $\mathcal{F}$ and $\mathcal{V}$ are denoted $\mathcal{T}(\mathcal{F},\mathcal{V})$. A binary relation on $\mathcal{T}(\mathcal{F},\mathcal{V})$ is a *rewrite relation* if it is compatible with $\mathcal{F}$-operations and closed under substitutions. The smallest rewrite relation that contains $\mathcal{R}$ is denoted $\to_{\mathcal{R}}$. The reflexive-transitive closure of rewrite steps $\to_{\mathcal{R}}$ is denoted $\to_{\mathcal{R}}^*$; the transitive closure, $\to_{\mathcal{R}}^+$. For further details about rewriting, see the survey by the first author and Jouannaud [10] or one of the books [2,33].

In [10], the following rewrite system $\mathcal{H}$ was introduced as an encoding of the Hydra Battle:[6]

$$\mathsf{h}(\mathsf{e}(x),y) \to \mathsf{h}(\mathsf{d}(x,y),\mathsf{S}(y)) \tag{1}$$
$$\mathsf{d}(\mathsf{g}(0,0),y) \to \mathsf{e}(0) \tag{2}$$
$$\mathsf{d}(\mathsf{g}(x,y),z) \to \mathsf{g}(\mathsf{e}(x),\mathsf{d}(y,z)) \tag{3}$$
$$\mathsf{d}(\mathsf{g}(\mathsf{g}(x,y),0),\mathsf{S}(z)) \to \mathsf{g}(\mathsf{d}(\mathsf{g}(x,y),\mathsf{S}(z)),\mathsf{d}(\mathsf{g}(x,y),z)) \tag{4}$$
$$\mathsf{g}(\mathsf{e}(x),\mathsf{e}(y)) \to \mathsf{e}(\mathsf{g}(x,y)) \tag{5}$$

(To clarify the connection to the standard Hydra Battle, as presented in Sect. 2, we have swapped the original arguments of the symbols d and h and make use of the unary function symbol S instead of the original c.)

The idea was for $\mathsf{h}(x,n)$ to represent the nth stage of the battle, with Hydra current being x, and with g serving as `cons` and 0 as `nil`. Then, $\mathsf{d}(n,x)$ marks the position of Hercules' search for a head to chop off (n is the replication factor); d was also meant to perform the duplication (which is the rôle of f in the functional program described in Section 2.2). The d in the first argument of g on the right side of rule (4) forces that branch to get smaller, via rules (3) and (2), assuming that branch has a head dangling at its right edge. The symbol e is used to signal completion of the operation on a branch, and settles towards the root after replication. The system was designed to allow various sterile derivations, as well as the primary, battle one.

Unfortunately, rule (4) does not perform as advertised; the system does not simulate the standard Hydra Battle, as defined in Sect. 2.[7] To rectify this, the first author proposed (on Pierre Lescanne's rewriting list [26]) the following

[6] Some (lost) version of this rewrite system had been presented by the first author at the Interdisciplinary Conference on Axiomatic Systems, in Columbus, OH, on December 16, 1988.

[7] The originally intended system probably had $\mathsf{g}(\mathsf{d}(\mathsf{g}(x,y),\mathsf{S}(z)),\mathsf{d}(\mathsf{g}(\mathsf{g}(x,y),0),z))$ as the right-hand side of rule (4). Some additional changes are needed for it to be able to simulate the standard Hydra Battle. We do not discuss this version any further.

System $\mathcal{D}$, comprising six rules:

$$\mathsf{h}(\mathsf{e}(x), y) \to \mathsf{h}(\mathsf{d}(x, y), \mathsf{S}(y)) \tag{6}$$
$$\mathsf{d}(\mathsf{g}(0, x), y) \to \mathsf{e}(x) \tag{7}$$
$$\mathsf{d}(\mathsf{g}(x, y), z) \to \mathsf{g}(\mathsf{d}(x, z), \mathsf{e}(y)) \tag{8}$$
$$\mathsf{d}(\mathsf{g}(\mathsf{g}(0, x), y), 0) \to \mathsf{e}(y) \tag{9}$$
$$\mathsf{d}(\mathsf{g}(\mathsf{g}(0, x), y), \mathsf{S}(z)) \to \mathsf{g}(\mathsf{e}(x), \mathsf{d}(\mathsf{g}(\mathsf{g}(0, x), y), z)) \tag{10}$$
$$\mathsf{g}(\mathsf{e}(x), \mathsf{e}(y)) \to \mathsf{e}(\mathsf{g}(x, y)) \tag{11}$$

We will see below that the underlying semantics of the symbol g changed in the transition from $\mathcal{H}$ to $\mathcal{D}$. (Here again we have swapped the arguments of the symbols d and h and make use of the unary function symbol S instead of the original c.)

5.1 Faithfulness

It is not hard to see that, indeed, $\mathcal{D}$ faithfully represents the standard Hydra Battle. We define a mapping $\mathcal{O}\colon CNF \to \mathcal{T}(\mathcal{F}, \emptyset)$, where $\mathcal{F}$ is the set of function symbols in $\mathcal{D}$:

$$\mathcal{O}(\alpha) := \begin{cases} 0 & \text{if } \alpha = 0 \\ \mathsf{g}(\mathcal{O}(\gamma), 0) & \text{if } \alpha = \omega^\gamma \\ \mathsf{g}(\mathcal{O}(\gamma), \mathcal{O}(\beta)) & \text{if } \alpha = \beta + \omega^\gamma \end{cases}$$

Each configuration (α, n) of the game is encoded by a term $\mathsf{h}(\mathsf{e}(\mathcal{O}(\alpha)), \mathsf{S}^n(0))$.

Lemma 5. *Let $\alpha \in CNF$, $\alpha > 0$, $n \in \mathbf{N} \setminus \{0\}$. Then $\mathsf{h}(\mathsf{e}(\mathcal{O}(\alpha)), \mathsf{S}^n(0)) \to_{\mathcal{D}}^+ \mathsf{h}(\mathsf{e}(\mathcal{O}(\alpha_n)), \mathsf{S}^{n+1}(0))$.*

Proof. Due to the presence of the rule $\mathsf{h}(\mathsf{e}(x), y) \to \mathsf{h}(\mathsf{d}(x, y), \mathsf{S}(y))$ in $\mathcal{D}$, it suffices to verify that $\mathsf{d}(\mathcal{O}(\alpha), \mathsf{S}^n(0)) \to_{\mathcal{D}}^+ \mathsf{e}(\mathcal{O}(\alpha_n))$. We proceed by induction on α.

1. Case $\alpha = \beta + 1$: By definition, $\alpha_n = \beta$ and $\mathcal{O}(\alpha) = \mathsf{g}(0, \mathcal{O}(\beta))$. Let $t = \mathcal{O}(\alpha)$ and $s = \mathcal{O}(\beta)$. To establish $\mathsf{d}(\mathcal{O}(\alpha), \mathsf{S}^n(0)) \to_{\mathcal{D}}^+ \mathsf{e}(\mathcal{O}(\alpha_n))$, we only need one rewrite step by rule (7):
$$\mathsf{d}(\mathsf{g}(0, s), \mathsf{S}^n(0)) \to_{\mathcal{D}} \mathsf{e}(s)\ .$$
2. Case $\alpha = \beta + \omega^{\gamma+1}$: By definition, $\alpha_n = \beta + \omega^\gamma \cdot n$ and $t = \mathcal{O}(\alpha) = \mathsf{g}(\mathsf{g}(0, \mathcal{O}(\gamma)), \mathcal{O}(\beta))$. Let $s = \mathcal{O}(\beta)$, $r = \mathcal{O}(\gamma)$. Then
$$\mathcal{O}(\alpha_n) = \underbrace{\mathsf{g}(r, \mathsf{g}(r, \cdots \mathsf{g}(r, s) \cdots))}_{n \text{ occurrences of } r}\ ,$$
and the following rewrite sequence suffices:
$$\begin{aligned} \mathsf{d}(\mathsf{g}(\mathsf{g}(0, r), s), \mathsf{S}^n(0)) &\to_{\mathcal{D}}^+ \mathsf{g}(\mathsf{e}(r), \mathsf{g}(\mathsf{e}(r), \cdots \mathsf{d}(\mathsf{g}(\mathsf{g}(0, r), s), 0) \cdots)) && (10)^n \\ &\to_{\mathcal{D}} \mathsf{g}(\mathsf{e}(r), \mathsf{g}(\mathsf{e}(r), \cdots \mathsf{g}(\mathsf{e}(r), \mathsf{e}(s)) \cdots)) && (9) \\ &\to_{\mathcal{D}}^+ \mathsf{e}(\mathsf{g}(r, \mathsf{g}(r, \cdots \mathsf{g}(r, s) \cdots)))\ . && (11)^n \end{aligned}$$

3. Case $\alpha = \beta + \omega^\gamma$, $\gamma \in \mathsf{Lim}$: By definition, $\alpha_n = \beta + \omega^{\gamma_n}$ and $\mathcal{O}(\alpha) = \mathsf{g}(\mathcal{O}(\gamma), \mathcal{O}(\beta))$. Let $t = \mathcal{O}(\alpha)$, $s = \mathcal{O}(\beta)$, $r = \mathcal{O}(\gamma)$, and $u = \mathcal{O}(\gamma_n)$. By the induction hypotheses (IH), $\mathsf{d}(r, \mathsf{S}^n(0)) \to_{\mathcal{D}}^+ \mathsf{e}(u)$ holds. Hence, the following rewrite sequence suffices:

$$\begin{aligned} \mathsf{d}(\mathsf{g}(r,s), \mathsf{S}^n(0)) &\to_{\mathcal{D}} \mathsf{g}(\mathsf{d}(r, \mathsf{S}^n(0)), \mathsf{e}(s)) && (8)\\ &\to_{\mathcal{D}}^+ \mathsf{g}(\mathsf{e}(u), \mathsf{e}(s)) && (\mathrm{IH})\\ &\to_{\mathcal{D}} \mathsf{e}(\mathsf{g}(u,s)) \,. && (11) \end{aligned}$$

□

6 Termination

We will employ "reduction orders" to prove termination of the Hydra systems.

6.1 Reduction Orders

A rewrite system $\mathcal{R}$ and a partial order $\succ$ are *compatible* if $\mathcal{R} \subseteq \succ$. A rewrite relation that is also a well-founded partial order is called a *reduction order*. It is easy to see that an interpretation-based term order $\succ_{\mathcal{A}}$ is a reduction order if the algebra $(\mathcal{A}, \succ)$ is well-founded and monotone. We say that $(\mathcal{A}, \succ)$ is compatible with a rewrite system $\mathcal{R}$ if $\succ_{\mathcal{A}}$ is compatible with $\mathcal{R}$.

A system $\mathcal{R}$ is *terminating* if no infinite sequence of rewrite steps exists. Thus, $\mathcal{R}$ is terminating iff it is compatible with a reduction order $\succ$.

6.2 Termination Properties

The rules of System $\mathcal{D}$ are similar to the original proposed formalization of the Hydra battle. While the rules of System $\mathcal{H}$ defining h and g have been kept, the three rules defining d have been replaced by four rules.

As given, all but the first rule of $\mathcal{H}$ and $\mathcal{D}$ decrease in a simple recursive path order [5], with precedence $\mathsf{d} > \mathsf{g} > \mathsf{e}$. The difficulty is in arranging for the first argument of h to show a decrease, as well.

A terminating rewrite system is *simply terminating* if its termination can be proved by a reduction order, like the recursive path order, that enjoys the subterm property (namely, that subterms are smaller in the order).

Theorem 4. *System $\mathcal{H}$ is terminating, but not simply terminating.*

Proof. For now, we only prove the easy fact that $\mathcal{H}$ is not simply terminating, the termination proof is postponed to Sect. 8. To show that $\mathcal{H}$ is not simply terminating, note that the rewrite step

$$\mathsf{h}(\mathsf{e}(x), \mathsf{e}(x)) \to_{\mathcal{H}} \mathsf{h}(\mathsf{d}(x, \mathsf{e}(x)), \mathsf{S}(\mathsf{e}(x)))$$

leads to a term that has the initial term embedded (homeomorphically) within it. □

By the same token, $\mathcal{D}$ is not simply terminating.

Theorem 5. *System $\mathcal{D}$ is terminating, but not simply terminating.*

Again the proof of termination is deferred until Sect. 8.

6.3 Previous Problems

While termination of $\mathcal{H}$, and implicitly of $\mathcal{D}$, has been claimed a number of times in the literature, to our best knowledge no (full, correct) termination proof has been provided.

For example, consider the proof sketch in [7, p. 8]. The idea of the proof is to use a general path order [9] that employs the following interpretations of the function symbols in $\mathcal{H}$ into the ordinals:

$$\begin{aligned} [\![\mathsf{g}(x,y)]\!] &:= \omega^{[\![x]\!]} + [\![y]\!] & [\![\mathsf{h}(x,z)]\!] &:= [\![x]\!] + [\![z]\!] \\ [\![\mathsf{d}(x,z)]\!] &:= \mathsf{pred}_{[\![z]\!]}([\![x]\!]) & [\![\mathsf{e}(x)]\!] &:= [\![x]\!] \\ [\![\mathsf{S}(x)]\!] &:= [\![x]\!] + 1 & [\![0]\!] &:= 1\ . \end{aligned}$$

The operator pred_ζ is conceived as a suitable extension of the operator α_ζ for $\zeta < \omega$; that is, we can assume $\mathsf{pred}_n(\alpha) = \alpha_n$.

One prerequisite to employ a general path order successfully is that, for all ground instances $l\sigma \to r\sigma$ of rules in $\mathcal{H}$, $[\![l\sigma]\!] \geqslant [\![r\sigma]\!]$ holds. However, by definition, we have

$$[\![\mathsf{d}(0,0)]\!] = [\![\mathsf{d}]\!](1,1) = 1_1 = 0\ ,$$

and therefore

$$\begin{aligned} [\![\mathsf{d}(\mathsf{g}(\mathsf{d}(0,0),\mathsf{d}(0,0)),0)]\!] &= [\![d]\!]([\![\mathsf{g}]\!]([\![\mathsf{d}(0,0)]\!],[\![\mathsf{d}(0,0)]\!]),1) \\ &= [\![d]\!](1,1) \\ &= 0 < 1 = [\![\mathsf{g}(\mathsf{e}(\mathsf{d}(0,0)),\mathsf{d}(\mathsf{d}(0,0),0))]\!]\ . \end{aligned}$$

Unfortunately, $\mathsf{d}(\mathsf{g}(\mathsf{d}(0,0),\mathsf{d}(0,0)),0) \to \mathsf{g}(\mathsf{e}(\mathsf{d}(0,0)),\mathsf{d}(\mathsf{d}(0,0),0))$ is an instance of rule (3).

Although this problem can be relatively easily rectified, there is a more serious problem with the proposed interpretation $[\![\cdot]\!]$. This interpretation is employed as one of the component functions of the general path order; to infer termination, these component functions (and hence the interpretation $[\![\cdot]\!]$) should be weakly monotone.

However, the interpretation function $[\![d]\!]$ is not weakly monotone in its first argument: Consider two hydræ $a = \mathsf{g}(0,\mathsf{d}(0,0))$ and $b = \mathsf{g}(\mathsf{d}(0,0),\mathsf{g}(\mathsf{d}(0,0),\mathsf{g}(\mathsf{d}(0,0),0)))$ with ordinal values $[\![a]\!] = \omega$ and $[\![b]\!] = 4$, respectively. Clearly $\omega > 4$ in the usual comparison of (set-theoretic) ordinals. But,

$$[\![\mathsf{d}(a,0)]\!] = [\![\mathsf{d}]\!](\omega,1) = \mathsf{pred}_1(\omega) = 1 < 3 = \mathsf{pred}_1(4) = [\![\mathsf{d}]\!](4,1) = [\![\mathsf{d}(b,0)]\!]\ .$$

Strictly speaking, one only needs monotonicity for terms that can rewrite to each other, that is, $f_{\mathcal{A}}(\ldots x \ldots) > f_{\mathcal{A}}(\ldots y \ldots)$ when $x > y$ *and* $x \to_{\mathcal{R}} y$; cf. [9, Thm. 2]. (Here $\mathcal{A}$ denotes an $\mathcal{F}$-algebra.) But consider rule (4) instantiated as follows:

$$x = \mathsf{d}(\mathsf{g}(\mathsf{g}(0,\mathsf{d}(0,0)),0),\mathsf{S}(b)) \to \mathsf{g}(\mathsf{d}(\mathsf{g}(0,\mathsf{d}(0,0)),\mathsf{S}(b)),\mathsf{d}(\mathsf{g}(0,\mathsf{d}(0,0)),b)) = y \ ,$$

where b is defined as above. Then $[\![\mathsf{g}(\mathsf{g}(0,\mathsf{d}(0,0)),0)]\!] = [\![\mathsf{g}]\!](\omega,1) = \omega^\omega + 1$ with $[\![\mathsf{g}(0,\mathsf{d}(0,0))]\!] = \omega$. Hence $[\![x]\!] = [\![\mathsf{d}]\!](\omega^\omega + 1, 5) = \omega^\omega > \omega^5 + 4 = [\![\mathsf{g}]\!](\mathsf{pred}_5(\omega), \mathsf{pred}_4(\omega)) = [\![y]\!]$ and thus *both* assumptions are fulfilled with respect to x and y; unfortunately $\mathsf{pred}_1(\omega^\omega) = \omega < \omega^5 + 3 = \mathsf{pred}_1(\omega^5 + 4)$. Proceeding in the same way as above, we again derive a counterexample.

To overcome this problem, we introduce (in the next section) a notation system for ordinals and use it, instead, as the domain of our interpretation functions.

7 In Preparation

Following an approach taken by Gaisi Takeuti [32], we introduce an alternate notation for ordinals below ϵ_0. This notation will enjoy the desired weak monotonicity property. We define a subset OT of terms over the signature $\{\omega, +\}$. (The function symbol ω is unary; the symbol $+$ is varyadic.) We write ω^α for $\omega(\alpha)$. In the definition of OT, we make use of an auxiliary subset $\mathsf{P} \subset \mathsf{OT}$.

Definition 3. *The definition of* OT *and* P *proceeds by mutual induction:*

1. $0 \in \mathsf{OT}$
2. *If* $\alpha_1, \ldots, \alpha_m \in \mathsf{P}$*, then* $\alpha_1 + \cdots + \alpha_m \in \mathsf{OT}$*.*
3. *If* $\alpha \in \mathsf{OT}$*, then* $\omega^\alpha \in \mathsf{P}$*, and* $\omega^\alpha \in \mathsf{OT}$*.*

The elements of OT *are called* ordinal terms *and are denoted by lower-case Greek letters. If no confusion can arise, we simply speak of* ordinals*.*

To simplify reading, we abbreviate the term ω^0 by 1. For the remainder of this section, the expression "ordinal" will always refer to an element of OT, unless stated otherwise.

It follows from the definition of the set OT that any object in OT different from 0 can be written in the following form:

$$\omega^{\alpha_1} + \omega^{\alpha_2} + \cdots + \omega^{\alpha_n} \ , \tag{12}$$

where each of the $\alpha_1, \ldots, \alpha_n$ has the same property.[8] However, due to the above definition, $\alpha + 0$ is *not* an ordinal, as $0 \notin \mathsf{P}$. To cure this, we introduce a binary operation $+$ on OT: Let $\alpha, \beta \in \mathsf{OT}$ be of form $\omega^{\alpha_1} + \cdots + \omega^{\alpha_n}$, $\beta = \omega^{\beta_1} + \cdots + \omega^{\beta_m}$. Then $\alpha + \beta$ is defined as $\omega^{\alpha_1} + \cdots + \omega^{\alpha_n} + \omega^{\beta_1} + \cdots + \omega^{\beta_m}$. Otherwise, we define $\alpha + 0 = 0 + \alpha = \alpha$. We will not distinguish between the binary operation $+$ and its varyadic rendering.

[8] This would not hold had we defined OT to be the set of terms over the signature $\{\omega, +\}$.

Definition 4 (Takeuti [32]). *We inductively define an equivalence $\sim$ and a partial order $\succ$ so that they satisfy the following clauses:*

1. *0 is the minimal element of $\succ$.*
2. *For $\alpha \in \mathsf{OT}$ of form (12), assume α contains two consecutive terms ω^{α_i} and $\omega^{\alpha_{i+1}}$ with $\alpha_{i+1} \succ \alpha_i$. So, α has the form*

$$\cdots + \omega^{\alpha_i} + \omega^{\alpha_{i+1}} + \dots \ .$$

Let β be obtained by removing the expression "$\omega^{\alpha_i} +$ " from α, so that β is of the form

$$\cdots + \omega^{\alpha_{i+1}} + \dots \ .$$

Then $\alpha \sim \beta$.
3. *Suppose $\alpha = \omega^{\alpha_1} + \cdots + \omega^{\alpha_m}$, $\beta = \omega^{\beta_1} + \cdots + \omega^{\beta_n}$, $\alpha_1 \succcurlyeq \alpha_2 \succcurlyeq \cdots \succcurlyeq \alpha_m$, and $\beta_1 \succcurlyeq \beta_2 \succcurlyeq \cdots \succcurlyeq \beta_n$, hold. ($\alpha \succcurlyeq \beta$ means $\alpha \succ \beta$ or $\alpha \sim \beta$.) Then, $\alpha \succ \beta$ if either $\alpha_i \succ \beta_i$ for some $i \in [1, m]$ and $\alpha_j \sim \beta_j$ for all $j \in [1, i-1]$, or $m > n$ and $\alpha_i \sim \beta_i$ holds for all $i \in [1, n]$.*

Remark 3. Note that ordinal addition $+$ is not commutative, not even up to the equivalence $\sim$, as we have $1 + \omega \sim \omega \not\sim \omega + 1$. □

We can identify the natural numbers $\mathbf{N}$ with the ordinals less than ω, as the usual comparison of natural numbers coincides with the above partial order $\succ$ on ordinal terms less than ω. So, we freely write $1 + 1$ as 2, $1 + 1 + 1$ as 3, and so on. By definition, for any $\alpha > 0$ in OT, there exists a unique $\beta \in \mathsf{OT}$ with $\alpha \sim \beta$ so that β can be written as

$$\omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_n} \qquad \text{with } \beta_1 \succcurlyeq \cdots \succcurlyeq \beta_n \ , \tag{13}$$

where $\beta_1 \succcurlyeq \cdots \succcurlyeq \beta_n$. If β is written in this way, we say that it is in *normal-form*. The set of all ordinal terms in normal-form together with 0 is denoted NF. The unique normal-form of a given ordinal term α is denoted $\mathsf{NF}(\alpha)$.

Remark 4. Note that our definition of the ordinal notation system OT is non-standard. Usually one identifies $\alpha \in \mathsf{OT}$ and its normal-form $\mathsf{NF}(\alpha)$ and instead of $\sim$ simply the equality $=$ is written.

Any $\alpha \in \mathsf{NF}$ uniquely represents a set-theoretic ordinals in CNF. The following lemma is immediate:

Lemma 6.

1. *The relation $\succ$ is a linear partial order on NF.*
2. *The relation $\succ$ is well-founded and* $\mathrm{otype}(\succ) = \epsilon_0$.

We extend the well-founded, linear order $\succ$ on NF to a well-founded, partial order $\succ$ on OT. To simplify notation we denote the extended relation with the same symbol, no confusion will arise from this. For $\alpha, \beta \in \mathsf{OT}$ define: $\alpha \succ \beta$, if $\mathsf{NF}(\alpha) \succ \mathsf{NF}(\beta)$. It follows that $\succ$ is a partial order and that $\alpha \succcurlyeq \beta \succ \gamma$ and $\alpha \succ \beta \succcurlyeq \gamma$ each imply $\alpha \succ \gamma$. The next lemma is a direct consequence of the definitions; we essentially employ the fact that $\mathsf{NF}(\alpha + \beta) = \mathsf{NF}(\mathsf{NF}(\alpha) + \mathsf{NF}(\beta))$.

Lemma 7. *Let* $\alpha, \beta, \gamma \in \mathsf{OT}$.

1. $\alpha + \beta \succcurlyeq \alpha, \beta$.
2. $\omega^\alpha \succ \alpha$.
3. *If* $\alpha \succ \beta$, *then* $\omega^\alpha \succ \omega^\beta$.
4. *If* $\alpha \succ \beta$, *then* $\gamma + \alpha \succ \gamma + \beta$ *and* $\alpha + \gamma \succcurlyeq \beta + \gamma$.
5. *If* $\alpha \in \mathsf{P}$ *and* $\alpha \succ \beta, \gamma$, *then* $\alpha \succ \beta + \gamma$.

The central idea of the above notation system is the separation of the identity of ordinal terms (denoted by $=$) and the identity of their set-theoretic counterparts (denoted by $\sim$). We will see in the next section that this pedantry is essential for a successful definition of the interpretation functions.

Based on $\succ$ and $\sim$, we define a partial order $\sqsupset$ and an equivalence relation $\equiv$ on OT. We write $\mathsf{N}(\alpha)$ to denote the number of occurrences of ω in α. Note that $\mathsf{N}(n) = n$ for any natural number n, since $1 = \omega^0$.

Definition 5. *Let* $\alpha, \beta \in \mathsf{OT}$. *We set:*

1. $\alpha \sqsupset \beta$ *if* $\alpha \succ \beta$, $\mathsf{N}(\alpha) \geqslant \mathsf{N}(\beta)$ *or* $\alpha \sim \beta$, $\mathsf{N}(\alpha) > \mathsf{N}(\beta)$ *and*
2. $\alpha \equiv \beta$ *if* $\alpha \sim \beta$, $\mathsf{N}(\alpha) = \mathsf{N}(\beta)$.

Define the quasi-order $\sqsupset^\equiv$: $\alpha \sqsupset^\equiv \beta$, *if* $\alpha \ (\sqsupset \cup \equiv) \ \beta$.

Example 3. Consider $\omega + \omega^2$ and $\omega + 3$. Then $\omega + \omega^2 \sqsupset \omega + 3$, as $\mathsf{NF}(\omega + \omega^2) = \omega^2 \succ \omega + 3 = \mathsf{NF}(\omega + 3)$ and $\mathsf{N}(\omega + \omega^2) = 5 = \mathsf{N}(\omega + 3)$. On the other hand, $\omega^2 \not\sqsupset \omega + 3$ as $\mathsf{N}(\omega + 3) = 5 > 3 = \mathsf{N}(\omega^2)$. □

This example shows that the relation $\sim$ is not compatible with the strict order $\sqsupset$.

Lemma 8. *The binary relation* $\sqsupset$ *is a well-founded order and* $\mathrm{otype}(\sqsupset) \leq \epsilon_0$. *Furthermore, for all* $n, m \in \mathbf{N}$, $n \sqsupset m$ *iff* $n > m$.

Proof. That $\sqsupset$ is a partial order is immediate from the definition. To verify that $\sqsupset$ is well-founded with $\mathrm{otype}(\sqsupset) \leq \epsilon_0$, it suffices to define an embedding $o\colon \mathsf{OT} \to \mathrm{CNF}$: $o(\alpha) := \omega^{\mathsf{NF}(\alpha)} + \mathsf{N}(\alpha)$. By case analysis on the definition of $\sqsupset$, one verifies that for all $\alpha, \beta \in \mathsf{OT}$, $\alpha \sqsupset \beta$ implies $o(\alpha) > o(\beta)$. Assume first that $\alpha \succ \beta$ and $\mathsf{N}(\alpha) \geqslant \mathsf{N}(\beta)$. Then, $\omega^{\mathsf{NF}(\alpha)} + \mathsf{N}(\alpha) > \omega^{\mathsf{NF}(\beta)} + \mathsf{N}(\beta)$ is immediate from the definition of the comparison $>$ of set-theoretic ordinals. Now assume $\alpha \sim \beta$ and $\mathsf{N}(\alpha) > \mathsf{N}(\beta)$. Then, $\omega^{\mathsf{NF}(\alpha)} + \mathsf{N}(\alpha) > \omega^{\mathsf{NF}(\beta)} + \mathsf{N}(\beta)$ follows similarly.

The second half of the lemma is a direct result of the definition of $\sqsupset$ and the definition of N. □

The following is again a direct consequence of the definitions:

Lemma 9. *Let* $\alpha, \beta, \gamma \in \mathsf{OT}$.

1. *If* $\alpha \sqsupset \beta$, *then* $\omega^\alpha \sqsupset \omega^\beta$.
2. *If* $\alpha \sqsupset \beta$, *then* $\gamma + \alpha \sqsupset \gamma + \beta$ *and* $\alpha + \gamma \sqsupset^\equiv \beta + \gamma$.
3. $\alpha + \beta \sqsupset^\equiv \alpha, \gamma$.

4. $\omega^\alpha \sqsupset \alpha$.

Let $p\colon \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ denote a fixed polynomial, strictly monotone in each argument.

Definition 6 (Predecessor). *We define the set of* n-predecessors of α induced by p. *Let* $\alpha \in \mathsf{OT}$. *Then*

$$\alpha[n] := \{\beta \mid \alpha \succ \beta \text{ and } p(\mathsf{N}(\alpha), n) \geqslant \mathsf{N}(\beta)\} \ .$$

The notion of an n-predecessor stems from [12]. However, we follow the idea of norm-based fundamental sequences; cf. [4].

Lemma 10. *Let* $\alpha \in \mathsf{OT}$ *and let* δ *denote a* $\sqsupset$*-maximal element of* $\alpha[n]$.

1. *The set* $\alpha[n]$ *is finite.*
2. *For each* $\beta \in \alpha[n]$: $\delta \sqsupset^{\equiv} \beta$.

Proof. The first assertion is trivial. For the second, observe that it follows from the definition of δ that for all $\beta \in \alpha[n]$, either $\delta \sqsupset \beta$, $\beta \equiv \delta$, or β and δ are incomparable with respect to $\sqsupset$. We prove that the last case can never happen. We assume $\alpha > 0$, as otherwise the assertion follows trivially. Let $\beta \in \alpha[n]$ be arbitrary but fixed, so that β, δ are incomparable.

The ordinals β and δ can only be incomparable if either of the following cases holds: (i) $\delta \prec \beta$ and $\mathsf{N}(\beta) < \mathsf{N}(\delta)$, or (ii) $\delta \succ \beta$ and $\mathsf{N}(\beta) > \mathsf{N}(\delta)$. As the cases are dual, it suffices to consider the first one. Assume $\beta \in \mathbf{N}$, then $\delta \in \mathbf{N}$ and $\mathsf{N}(\beta) = \beta \succ \delta = \mathsf{N}(\delta)$, which contradicts the assumption $\mathsf{N}(\delta) > \mathsf{N}(\beta)$. Hence, we can assume $\beta \succcurlyeq \omega$.

We define an ordinal term β^* as follows: $\beta^* := (\mathsf{N}(\delta) - \mathsf{N}(\beta)) + \beta$. As $\beta \succcurlyeq \omega$, $\beta^* \sim \beta$ holds. Furthermore, $\mathsf{N}(\beta^*) = \mathsf{N}(\delta) > \mathsf{N}(\beta)$, as $\mathsf{N}(\beta^*) = (\mathsf{N}(\delta) - \mathsf{N}(\beta)) + \mathsf{N}(\beta) = \mathsf{N}(\delta)$. So, $\beta^* \sqsupset \beta$. We show that $\beta^* \in \alpha[n]$: $\alpha \succ \beta \sim \beta^*$ implies $\alpha \succ \beta^*$. And $p(\mathsf{N}(\alpha), n) \geqslant \mathsf{N}(\delta) = \mathsf{N}(\beta^*)$ implies $p(\mathsf{N}(\alpha), n) \geqslant \mathsf{N}(\beta^*)$. We derive a contradiction to the assumption that δ is $\sqsupset$-maximal. □

By the above lemma a $\sqsupset$-maximal element of $\alpha[n]$ is, up to the equivalence $\equiv$, unique. In the following, for each $\alpha \in \mathsf{OT}$ and each $n \in \mathbf{N}$, we fix an arbitrary $\sqsupset$-maximal element and denote it with $P_n(\alpha)$.

Lemma 11. *Let* $\alpha \in \mathsf{OT}$ *and suppose* $\alpha \succcurlyeq \omega$. *Then* $\mathsf{N}(P_n(\alpha)) = p(\mathsf{N}(\alpha), n)$.

Proof. The proof follows the pattern of the proof of the previous lemma. □

The following lemma explains why the pedantry in the definition of the set of ordinal terms OT and the given definition of the partial order $\sqsupset$ is necessary:

Lemma 12. *Let* $\alpha, \beta \in \mathsf{OT}$, $n \in \mathbf{N}$.

1. *If* $\alpha, \beta > 0$ *and* $\alpha \sqsupset \beta$, *then* $P_n(\alpha) \sqsupset P_n(\beta)$.
2. *If* $\alpha \equiv \beta$, *then* $P_n(\alpha) \equiv P_n(\beta)$.
3. *Suppose* $m > n$. *Then* $P_m(\alpha) \sqsupset^{\equiv} P_n(\alpha)$.

We want to emphasize that the first property fails for the specific fundamental sequence $(\alpha_n)_{n\in\mathbf{N}}$ employed in the definition of the standard Hydra Battle; cf. Definition 1: We have $\omega > m$, but $\omega_n = n \not> m-1 = (m)_n$ for any $m > n$.

Proof (of the lemma). We only show the first point; the arguments for the other points are similar, but simpler. Assume $\alpha \sqsupset \beta$. First, we show the lemma for the special-case, where $\alpha \in \mathbf{N}$. This assumption implies $\beta \in \mathbf{N}$. Hence, $P_n(\alpha) = \alpha - 1 \sqsupset \beta - 1 = P_n(\beta)$. Consider the case $\alpha \succcurlyeq \omega$. We proceed by cases, according to the definition of $\sqsupset$:

1. Subcase $\alpha \succ \beta$ and $\mathsf{N}(\alpha) \geqslant \mathsf{N}(\beta)$: Then monotonicity of p implies that $p(\mathsf{N}(\alpha), n) \geqslant \mathsf{N}(\beta)$ holds. Thus, $\beta \in \alpha[n]$. By Lemma 10(2), we conclude $P_n(\alpha) \succcurlyeq \beta \succ P_n(\beta)$, which implies $P_n(\alpha) \succ P_n(\beta)$. By Lemma 11, we get: $\mathsf{N}(P_n(\alpha)) = p(\mathsf{N}(\alpha), n) \geqslant p(\mathsf{N}(\beta), n) \geqslant \mathsf{N}(P_n(\beta))$. In summary, we see $P_n(\alpha) \sqsupset P_n(\beta)$.
2. Subcase $\alpha \sim \beta$ and $\mathsf{N}(\alpha) > \mathsf{N}(\beta)$: From the assumptions we conclude $P_n(\beta) \in \alpha[n]$, as $\alpha \sim \beta \succ P_n(\beta)$ and $p(\mathsf{N}(\alpha), n) > p(\mathsf{N}(\beta), n) \geqslant \mathsf{N}(P_n(\beta))$. Hence, Lemma 10 implies $P_n(\alpha) \sqsupset P_n(\beta)$ or $P_n(\alpha) \equiv P_n(\beta)$. If the former case holds, the lemma is established. Assume the latter. By definition of $\equiv$, we see that $\mathsf{N}(P_n(\alpha)) = \mathsf{N}(P_n(\beta))$. On the other hand, we obtain: $\mathsf{N}(P_n(\alpha)) = p(\mathsf{N}(\alpha), n) > p(\mathsf{N}(\beta), n) \geqslant \mathsf{N}(P_n(\beta))$. We have derived a contradiction.

□

8 Termination

The purpose of this section is to prove Theorem 5. Based on the construction given below, it is easy to see how to also prove Theorem 4; hence, we leave that one to the reader.

8.1 Interpretation

Using the ordinal notation of the previous section, the termination proof is relatively simple. Let $\mathcal{F}$ denote the signature of System $\mathcal{D}$. We define the $\mathcal{F}$-algebra $(\mathcal{A}, \rhd)$ and provide a proof that $\mathcal{A}$ is well-founded, which is easy, but – more significantly – $\mathcal{A}$ is weakly monotone. The domain of $\mathcal{A}$ is the set

$$\{(\alpha, 1) \mid \alpha \in \mathsf{OT}\} \cup \{(0,0)\} \,.$$

We define the quasi-order $\trianglerighteq$ on the pairs as follows:

$$(\alpha, a) \trianglerighteq (\beta, b) \quad \text{iff} \quad (\alpha \sqsupset^{\equiv} \beta \wedge a = b = 1) \text{ or } (\alpha \sqsupset^{\equiv} \beta \wedge a > b) \,.$$

The following operations interpret the elements of $\mathcal{F}$:

$$\begin{aligned}
\mathsf{d}_{\mathcal{A}} :&\quad (\alpha, a), (\beta, b) \mapsto (P_{\mathsf{N}(\beta)}(\alpha), 1) && \alpha \neq 0\\
&\quad (0, a), (\beta, b) \mapsto (0, 0)\\
\mathsf{h}_{\mathcal{A}} :&\quad (\alpha, a), (\beta, b) \mapsto (0, 0)\\
\mathsf{g}_{\mathcal{A}} :&\quad (\alpha, 1), (\beta, b) \mapsto (\beta + \omega^{\alpha}, 1)\\
&\quad (0, 0), (\beta, b) \mapsto (0, 0)\\
\mathsf{e}_{\mathcal{A}} :&\quad (\alpha, a) \mapsto (\alpha, 1)\\
\mathsf{S}_{\mathcal{A}} :&\quad (\alpha, a) \mapsto (\alpha + 1, 1)\\
0_{\mathcal{A}} :&\quad (0, 1)
\end{aligned}$$

Define the strict order $\blacktriangleright$ by replacing $\sqsupset^{\equiv}$ by $\succ$ in the above definition. The orders $\trianglerighteq$ and $\blacktriangleright$ naturally extend to terms, denoted $\trianglerighteq_{\mathcal{A}}$ and $\blacktriangleright_{\mathcal{A}}$, respectively. Fix the parameter in the definition of n-predecessors:

$$p(m, n) := (m + 1) \cdot (n + 1) \,.$$

Let $\triangleright$ denote the partial order induced by the quasi-order $\trianglerighteq$. With the help of Lemma 12, the following is not difficult to prove.

Lemma 13. *The $\mathcal{F}$-algebra $(\mathcal{A}, \triangleright)$ is weakly monotone and well-founded.*

Lemma 14. *For each rule $l \to r$ in $\mathcal{D}$, we have $l \trianglerighteq_{\mathcal{A}} r$, that is, $\mathcal{A}$ is a quasi-model of $\mathcal{D}$.*

Proof. We consider only the rules (8) and (10), as it is easy to check the properties for the other rules.

1. Case $\mathsf{d}(\mathsf{g}(x, y), z) \to \mathsf{g}(\mathsf{d}(x, z), \mathsf{e}(y))$: We have to show

   $$\mathsf{d}_{\mathcal{A}}(\mathsf{g}_{\mathcal{A}}((\alpha, a), (\beta, b)), (\gamma, c)) \trianglerighteq \mathsf{g}_{\mathcal{A}}(\mathsf{d}_{\mathcal{A}}((\alpha, a), (\gamma, c)), \mathsf{e}_{\mathcal{A}}((\beta, b))) \,.$$

   One of the following subcases holds (i) $\alpha > 0$ (ii) $\alpha = 0$. We may assume subcase (i) holds. Assume otherwise; then it is not hard to see that the right-hand side of the above equation rewrites to $(0, 0)$. From this the claim follows easily.
   Accordingly, we obtain

   $$\begin{aligned}
   \mathsf{d}_{\mathcal{A}}(\mathsf{g}_{\mathcal{A}}((\alpha, 1), (\beta, b)), (\gamma, c)) &= (P_n(\beta + \omega^{\alpha}), 1) \trianglerighteq\\
   &\trianglerighteq (\beta + \omega^{P_n(\alpha)}, 1) = \mathsf{g}_{\mathcal{A}}(\mathsf{d}_{\mathcal{A}}((\alpha, 1), (\gamma, c)), \mathsf{e}_{\mathcal{A}}((\beta, b))) \,,
   \end{aligned}$$

for $n = \mathsf{N}(\gamma)$. We have to show that $P_n(\beta+\omega^\alpha) \sqsupset^{\equiv} \beta+\omega^{P_n(\alpha)}$. By Lemma 7, we obtain $\beta + \omega^\alpha \succ \beta + \omega^{P_n(\alpha)}$. By definition of the polynomial p and the norm-function N, and Lemma 12 it suffices to observe:

$$\begin{aligned}(\mathsf{N}(\beta + \omega^\alpha) + 1)(n+1) = (\mathsf{N}(\beta) + \mathsf{N}(\alpha) + 2)(n+1) \geqslant \\ \geqslant \mathsf{N}(\beta) + 1 + (\mathsf{N}(\alpha) + 1)(n+1) \geqslant \mathsf{N}(\beta + \omega^{P_n(\alpha)}) \ .\end{aligned}$$

2. Case $\mathsf{d}(\mathsf{g}(\mathsf{g}(0,x),y),\mathsf{S}(z)) \to \mathsf{g}(\mathsf{e}(x),\mathsf{d}(\mathsf{g}(\mathsf{g}(0,x),y),z))$: We show

$$\begin{aligned}\mathsf{d}_{\mathcal{A}}(\mathsf{g}_{\mathcal{A}}(\mathsf{g}_{\mathcal{A}}(0_{\mathcal{A}},(\alpha,a)),(\beta,b)),\mathsf{S}_{\mathcal{A}}((\gamma,c))) \trianglerighteq \\ \trianglerighteq \mathsf{g}_{\mathcal{A}}(\mathsf{e}_{\mathcal{A}}((\alpha,a)),\mathsf{d}_{\mathcal{A}}(\mathsf{g}_{\mathcal{A}}(\mathsf{g}_{\mathcal{A}}(0_{\mathcal{A}},(\alpha,a)),(\beta,b)),(\gamma,c))) \ ,\end{aligned}$$

for all $(\alpha,a),(\beta,b),(\gamma,c) \in \mathcal{A}$. By definition, the left-hand side rewrites to

$$(P_{\mathsf{N}(\gamma+1)}(\beta + \omega^{\alpha+1}), 1) \ ,$$

while the right side becomes

$$(P_{\mathsf{N}(\gamma)}(\beta + \omega^{\alpha+1}) + \omega^\alpha, 1) \ ,$$

and we have to show $P_{\mathsf{N}(\gamma)+1}(\beta + \omega^{\alpha+1}) \sqsupset^{\equiv} P_{\mathsf{N}(\gamma)}(\beta + \omega^{\alpha+1}) + \omega^\alpha$. By Definition 6 and Lemma 7(5) we obtain:

$$\beta + \omega^{\alpha+1} \succ P_{\mathsf{N}(\gamma)}(\beta + \omega^{\alpha+1}) + \omega^\alpha \ .$$

Therefore, it suffices to show

$$(\mathsf{N}(\beta + \omega^{\alpha+1}) + 1)(\mathsf{N}(\gamma) + 2) \geqslant \mathsf{N}(P_{\mathsf{N}(\gamma)}(\beta + \omega^{\alpha+1}) + \omega^\alpha) \ ,$$

which follows by a simply calculation:

$$\begin{aligned}(\mathsf{N}(\beta + \omega^{\alpha+1}) + 1)(n+2) \geqslant (\mathsf{N}(\beta + \omega^{\alpha+1}) + 1)(n+1) + \mathsf{N}(\omega^{\alpha+1}) \geqslant \\ \geqslant \mathsf{N}(P_n(\beta + \omega^{\alpha+1}) + \omega^\alpha) \ ,\end{aligned}$$

with $n = \mathsf{N}(\gamma)$.

□

8.2 Dependencies

Finally, we are in position to prove Theorem 5, and employ a specific variant of the dependency-pair method of [1]. (This choice of method is not critical; equivalently, a proof by induction upto ϵ_0 could be given, or some other method employed.)

To keep this paper more-or-less self-contained, we first recall some basic definitions and lemmas. We write $\lhd$ to denote the proper subterm relation and $\trianglerighteq$ for (not necessarily proper) superterm. Let $\mathcal{R}$ be some rewrite system and denote the set of all minimal non-terminating terms by $\mathcal{T}_\infty$ (minimal in the sense of the subterm relation).

Lemma 15. *For every term $t \in \mathcal{T}_\infty$ there exist a rewrite rule $l \to r \in \mathcal{R}$, a substitution σ, and a non-variable subterm u of r, such that $t \xrightarrow{not\ top}{}^*_{\mathcal{R}} l\sigma \xrightarrow{top}_{\mathcal{R}} r\sigma \trianglerighteq u\sigma$ and $u\sigma \in \mathcal{T}_\infty$.*

By the lemma, it is not difficult to see that any term in $\mathcal{T}_\infty$ has a defined root symbol. This, we exploit in the next definition.

Let $\mathcal{R}$ be a rewriting system over a signature $\mathcal{F}$. Let $\widehat{f}$ denote a fresh function symbol with the same arity as $f \in \mathcal{F}$ and let $\widehat{t}$ denote $\widehat{f}(t_1, \ldots, t_n)$, for term $t = f(t_1, \ldots, t_n)$. The set $\mathsf{DP}(\mathcal{R})$ of *dependency pairs* is defined as follows:

$$\mathsf{DP}(\mathcal{R}) := \{\widehat{l} \to \widehat{u} \mid l \to r \in \mathcal{R}, r \trianglerighteq u \not\trianglelefteq l, \text{root of } u \text{ defined}\} .$$

The nodes of the *dependency graph* $\mathsf{DG}(\mathcal{R})$, for rewrite system $\mathcal{R}$, are the dependency pairs of $\mathcal{R}$ and there is an arrow from $s \to t$ to $u \to v$ if and only if there exist substitutions σ and ρ such that $t\sigma \to_{\mathcal{R}} u\rho$. A *dp-cycle* is a nonempty subset $\mathcal{C}$ of dependency pairs of $\mathsf{DP}(\mathcal{R})$ if for every two (not necessarily distinct) pairs $s \to t$ and $u \to v$ in $\mathcal{C}$ there exists a nonempty path in $\mathcal{C}$ between them. By the above lemma and employing the notion of dependency graph, nontermination of $\mathcal{R}$ implies the existence of an infinite sequence of the following form:

$$t_1 \to^*_{\mathcal{R}} t_2 \to_{\mathcal{C}} t_3 \to^*_{\mathcal{R}} t_4 \to_{\mathcal{C}} t_5 \cdots ,$$

where $t_i \in \{\widehat{t} \mid t \in \mathcal{T}_\infty\}$, $\mathcal{C} \subseteq \mathsf{DG}(\mathcal{R})$ and the rules in $\mathcal{C}$ are applied infinitely often. Such a sequence is called *$\mathcal{C}$-minimal*. Thus, to prove termination it suffices to verify that no such sequences can exist.

Theorem 6 (Arts & Giesl [1]). *A finite term-rewriting system $\mathcal{R}$ is terminating if no $\mathcal{C}$-minimal sequence exists for any dp-cycle in $\mathsf{DG}(\mathcal{R})$.*

An *argument filtering* is a mapping ρ that associates with every function symbol either an argument position or a list of argument positions. The signature $\mathcal{F}_\rho$ contains m-ary function symbols f^ρ for any $f \in \mathcal{F}$ with $\rho(f) = [i_1, \ldots, i_m]$. The mapping ρ naturally gives rise to a function $\rho\colon\ \mathcal{T}(\mathcal{F}, \mathcal{V}) \to \mathcal{T}(\mathcal{F}_\rho, \mathcal{V})$.

Theorem 7 (Arts, Giesl & Ohlebusch [16]). *Let $\mathcal{R}$ be a term-rewriting system and $\mathcal{C}$ be a dp-cycle in $\mathsf{DG}(\mathcal{R})$. If there exists an argument filtering and a* reduction pair *$(\gtrsim, >)$ such that $\rho(\mathcal{R}) \subseteq \gtrsim$, $\rho(\mathcal{C}) \subseteq \gtrsim \cup >$, and $\rho(\mathcal{C}) \cap > \neq \emptyset$, then there are no $\mathcal{C}$-minimal rewrite sequences.*

8.3 Reduction

The proof depends on the following:

Lemma 16. *The pair $(\trianglerighteq_{\mathcal{A}}, \blacktriangleright_{\mathcal{A}})$ forms a reduction pair.*

Proof. One has to show that $\trianglerighteq_{\mathcal{A}}$ is a quasi-order that is closed under $\mathcal{F}$-operations and substitutions, that $\blacktriangleright_{\mathcal{A}}$ is well-founded and closed under substitutions, and – finally – that $\trianglerighteq_{\mathcal{A}} \circ \blacktriangleright_{\mathcal{A}} \subseteq \blacktriangleright_{\mathcal{A}}$. The first two items follow

directly from the definitions. Therefore, we only have to verify that, for all $(\alpha, a), (\beta, b), (\gamma, c) \in \mathcal{A}$, if $(\alpha, a) \trianglerighteq (\beta, b) \blacktriangleright (\gamma, c)$, then also $(\alpha, a) \blacktriangleright (\gamma, c)$.

Without loss of generality, assume $a = b = c = 1$: Assume otherwise, then $a = b = c = 0$ is impossible, as $(\beta, 0) \blacktriangleright (\gamma, 0)$ cannot hold. Hence, the only possibility is $a = b = 1$ and $c = 0$. But, by definition of $\mathcal{A}$, this implies $\gamma = 0$ and clearly $(\alpha, 1) \blacktriangleright (0, 0)$.

Given that $a = b = c = 1$ holds, the assumption specializes to $\alpha \sqsupset^{\equiv} \beta \succ \gamma$. We proceed by case analysis on $\alpha \sqsupset^{\equiv} \beta$. Either $\alpha \succ \beta$ and $\mathsf{N}(\alpha) \geqslant \mathsf{N}(\beta)$ or $\alpha \sim \beta$ and $\mathsf{N}(\alpha) \geqslant \mathsf{N}(\beta)$. In both cases, $\alpha \succcurlyeq \beta$ holds. Hence, by transitivity of $\succ$, $\alpha \succ \gamma$ follows. □

Proof (of Theorem 5). Consider the dependency pairs of $\mathcal{D}$:

$$\widehat{\mathsf{h}}(\mathsf{e}(x), y) \to \widehat{\mathsf{h}}(\mathsf{d}(x, y), \mathsf{S}(y)) \tag{14}$$

$$\widehat{\mathsf{h}}(\mathsf{e}(x), y) \to \widehat{\mathsf{d}}(x, y) \tag{15}$$

$$\widehat{\mathsf{d}}(\mathsf{g}(\mathsf{g}(0, x), y), \mathsf{S}(z)) \to \widehat{\mathsf{g}}(\mathsf{e}(x), \mathsf{d}(\mathsf{g}(\mathsf{g}(0, x), y), z)) \tag{16}$$

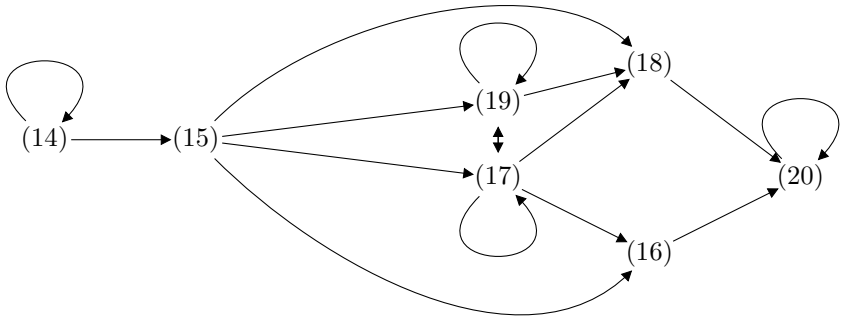
$$\widehat{\mathsf{d}}(\mathsf{g}(\mathsf{g}(0, x), y), \mathsf{S}(z)) \to \widehat{\mathsf{d}}(\mathsf{g}(\mathsf{g}(0, x), y), z) \tag{17}$$

$$\widehat{\mathsf{d}}(\mathsf{g}(x, y), z) \to \widehat{\mathsf{g}}(\mathsf{d}(x, z), \mathsf{e}(y)) \tag{18}$$

$$\widehat{\mathsf{d}}(\mathsf{g}(x, y), z) \to \widehat{\mathsf{d}}(x, z) \tag{19}$$

$$\widehat{\mathsf{g}}(\mathsf{e}(x), \mathsf{e}(y)) \to \widehat{\mathsf{g}}(x, y) \tag{20}$$

We construct the dependency graph $\mathsf{DG}(\mathcal{D})$:



The above interpretation extends to the extra dependency-pair functions $\widehat{f}$ as follows: We set $\widehat{f}_{\mathcal{A}}$ equal to $f_{\mathcal{A}}$, with the exception of $\widehat{\mathsf{h}}$, which we define via

$$\widehat{\mathsf{h}}_{\mathcal{A}}((\alpha, a), (\beta, b)) = (\alpha, a) \,.$$

Due to Theorems 6 and 7, it suffices to define suitable combinations of argument filterings and reduction pairs for cycles in $\mathsf{DP}(\mathcal{D})$. First, we consider the cycle $\{14\}$ and reduction pair $(\trianglerighteq_{\mathcal{A}}, \blacktriangleright_{\mathcal{A}})$. Due to Lemma 14, it remains to show that

$$\widehat{\mathsf{h}}(\mathsf{e}(x), y) \blacktriangleright \widehat{\mathsf{h}}(\mathsf{d}(x, y), \mathsf{S}(y)) \,. \tag{21}$$

Let $\mathbf{a}\colon \mathcal{V} \to A$ denote an arbitrary assignment with $[\mathbf{a}]_{\mathcal{A}}(x) = (\alpha, a)$, $[\mathbf{a}]_{\mathcal{A}}(y) = (\beta, b)$. If $\alpha > 0$, then (21) becomes $(\alpha, 1) \blacktriangleright (P_n(\alpha), 1)$, where $n = \mathsf{N}(\beta)$ and we have to show $\alpha \succ P_n(\alpha)$, which is a consequence of Definition 6. Assume otherwise $\alpha = 0$. Then (21) becomes $(0, 1) \trianglerighteq (0, 0)$, which follows from the definition of the relation $\blacktriangleright$.

With respect to the remaining dp-cycles, it is easy to see how a suitable combination of argument filterings and reduction pairs should be defined. In particular, note that these cycles can also be handled by applying the subterm criterion iteratively; cf. [20]. □

9 While Hydra Do

In this section, we convert the functional Hydra program $\mathcal{L}$ into an imperative, **while** program in stages. First, we replace each function with a similarly named procedure call, and the tail-recursive calls with iteration:

```
procedure H(x):
n := 0
while x ≠ nil do
    n := n + 1
    D(n, x)

procedure F(n, y, x):
for i := 1 to n do
    x := cons(y, x)
```

```
procedure D(n, x):
u := car(x)
if u = nil
then x := cdr(x)
else if car(u) = nil
        then F(n, cdr(u), cdr(x))
        else D(n, u)
             x := cons(u, cdr(x))
```

Using a stack s, implemented as a list (pushing via `cons`, popping via `car`), for the recursive calls to G, and combining all the procedures (x is the input hydra), we get:

```
n := 0
while x ≠ nil do
    n := n + 1
    u := car(x)
    if u = nil
    then x := cdr(x)
    else s := nil
         while car(u) ≠ nil do
             s := cons(s, cdr(x))
             x := u
             u := car(x)
         for i := 1 to n do
             x := cons(cdr(u), x)
         while s ≠ nil do
             x := cons(x, cdr(s))
             s := car(s)
```

It is easy to see that the inner loops all terminate. To show that the outer one does, one would need to show that x, qua ordinal, decreases with each outer iteration.

The list operations can be arithmetized by using a pairing function, such as $\texttt{cons}(x,y) := (x+y+1)^2+x$. Then $\texttt{nil} := 0$, $\texttt{car}(z) := z-\lfloor\sqrt{z}\rfloor^2$, and $\texttt{cdr}(z) := \lfloor\sqrt{z}\rfloor^2+\lfloor\sqrt{z}\rfloor-z-1$. (Any other set of pairing and projection functions would do just as well.) With this in mind, and with a tiny bit of algebraic manipulation, our final, wholly arithmetic, hard-to-prove-terminating **while** program is as follows:

$n := 0$
while $x > 0$ **do**
  $n := n + 1$
  $u := x - \lfloor\sqrt{x}\rfloor^2$
  if $u = 0$
  then $x := \lfloor\sqrt{x}\rfloor - 1$
  else $s := 0$
    while $u > \lfloor\sqrt{u}\rfloor^2$ **do**
      $s := s + (s + \lfloor\sqrt{x}\rfloor^2 + \lfloor\sqrt{x}\rfloor - x)^2$
      $x := u$
      $u := x - \lfloor\sqrt{x}\rfloor^2$
    for $i := 1$ **to** n **do**
      $x := (\lfloor\sqrt{u}\rfloor + x)^2 + \lfloor\sqrt{u}\rfloor - 1$
    while $s > 0$ **do**
      $x := x + (x + \lfloor\sqrt{s}\rfloor^2 + \lfloor\sqrt{s}\rfloor - s)^2$
      $s := s - \lfloor\sqrt{s}\rfloor^2$

Finally, the (integer-valued) truncated square-root $\lfloor\sqrt{z}\rfloor$ can be computed each time by a simple loop, searching for the largest integer whose square is no more than z:

$n := 0$
while $x > 0$ **do**
  $n := n + 1$
  $y := 0$; **while** $y^2 + 2y \le x$ **do** $y := y + 1$
  if $x = y^2$
  then $x := y - 1$
  else $s := 0$
    $r := 0$; **while** $r^2 + 2r \le x - y^2$ **do** $r := r + 1$
    while $x > y^2 + r^2$ **do**
      $y := 0$; **while** $y^2 + 2y \le x$ **do** $y := y + 1$
      $s := s + (s + y^2 + y - x)^2$
      $x := x - y^2$
      $r := 0$; **while** $r^2 + 2r \le x - y^2$ **do** $r := r + 1$
    for $i := 1$ **to** n **do** $x := r^2 + r - 1$
    while $s > 0$ **do**
      $r := 0$; **while** $r^2 + 2r \le s$ **do** $r := r + 1$
      $x := x + (x + r^2 + r - s)^2$
      $s := s - r^2$

Further simplifications are possible.

10 The Sky's the Limit

David Gries [18] has averred that for deterministic (or bounded nondeterministic) programs, since the number of steps of any terminating program is just some integer-valued function $t(\bar{x})$ that depends only on the program inputs $\bar{x}$, it is preferable to prove termination by showing "that each execution of the loop body decreases t by at least 1", than to use complicated well-founded orderings. This begs the issue, however, since the proof such a t exists for a program like Hydra requires transfinite induction up to ϵ_0, as we have seen above.

It is not hard to conjure up bigger battles, for example ones in which trees also grow in height. The following one – meant to require Γ_0 – is from [6]:

$$\begin{aligned}
G_n(\overline{x}) &\to G_{n+1}(p_n x)\\
p_n\langle x,y,z\rangle &\to \langle x,\overline{y},p_n z\rangle\\
p_{n+1}\langle A,y,z\rangle &\to \langle A,p_{n+1}y,r_n\langle B,\langle A,y,z\rangle,z\rangle\rangle\\
p_n\langle x,y,z\rangle &\to \overline{y}\\
p_n\langle B,y,z\rangle &\to r_n\langle B,y,z\rangle\\
r_{n+1}\langle B,y,z\rangle &\to \langle B,p_{n+1}y,r_n\langle B,y,z\rangle\rangle\\
r_n\langle x,y,z\rangle &\to \overline{z}\\
\langle x,\overline{y},\overline{z}\rangle &\to \overline{\langle x,y,z\rangle}
\end{aligned}$$

The A nodes are meant to act lexicographically; the B nodes, more like multisets. The bar acts like `e` of the Hydra system. Regarding the relevance to computer science of the (least) impredicative ordinal Γ_0, see [14].

Moreover, Γ_0 is by no means the end of the games. See [25] for rewrite systems that formalize the Hydra Battle up to the small Veblen ordinal, the maximal order type of the lexicographic path order [8]. Even larger hydras (so called Buchholz Hydræ) have been considered by Wilfried Buchholz [3].

References

1. Thomas Arts and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theor. Comput. Sci.*, 236:133–178, 2000.
2. Franz Baader and Tobias Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.
3. Wilfried Buchholz. An independence result for $(\Pi^1_1 - \mathrm{CA}) + \mathrm{BI}$. *Ann. Pure Appl. Logic*, 33:131–155, 1987.
4. Wilfried Buchholz, E. Adam Cichon and Andreas Weiermann. A uniform approach to fundamental sequences and hierarchies. *MLQ Math. Log. Q.*, 40:273–286, 1994.
5. Nachum Dershowitz. Orderings for term-rewriting systems. *Theor. Comput. Sci.*, 17(3):279–301, 1982.

6. Nachum Dershowitz. Trees, ordinals, and termination. *Proceedings of the Fourth International Joint Conference on Theory and Practice of Software Development (Orsay, France)*, M.-C. Gaudel and J.-P. Jouannaud, eds., Lecture Notes in Computer Science, vol. 668, pp. 243–250, Springer Verlag, 1993.
7. Nachum Dershowitz. 33 examples of termination. *French Spring School of Theoretical Computer Science. Advanced Course on Term Rewriting (Font Romeux, France, May 1993)*, H. Comon and J.-P. Jouannaud, eds., Lecture Notes in Computer Science, vol. 909, pp. 16–26, Springer Verlag, 1995.
8. Nachum Dershowitz and Mitsuhiro Okada. Proof-theoretic techniques for term rewriting theory. *Proceedings of the 3rd Annual Symposium on Logic in Computer Science*, IEEE, pp. 104–111, 1988.
9. Nachum Dershowitz and Charles Hoot. Natural termination, *Theor. Comput. Sci.*, 142(2):179–207, 1995.
10. Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. Chap. 6 in J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B: Formal Methods and Semantics*, pp. 245–319. Elsevier Science, 1990.
11. Nachum Dershowitz, Jean-Pierre Jouannaud and Jan Willem Klop. Open problems in rewriting. *Proceedings of the 4th International Conference on Rewriting Techniques and Applications*, Lecture Notes in Computer Science, vol. 488, pp. 445–456, Springer Verlag, 1991.
12. Matthew V. H. Fairtlough and Stanley S. Wainer. Hierarchies of provably recursive functions. In S. R. Buss, editor, *Handbook of Proof Theory*, pp. 149–207, Elsevier Science, 1998.
13. Rudolf Fleischer. Die another day. *Proceedings of the 4th International Conference 'FUN with Algorithms 4'*, Lecture Notes in Computer Science, vol. 4475, pp. 146–155, Springer Verlag, 2007. `http://www.cs.ust.hk/~rudolf/Paper/hydra_fun07.pdf`.
14. Jean H. Gallier. What's so special about Kruskal's Theorem and the ordinal Γ_0? A survey of some results in proof theory. *Ann. Pure Appl. Logic*, 53(3):199–260, 1991; Erratum, *Ann. Pure Appl. Logic* 89(2–3):275, 1997. `http://handle.dtic.mil/100.2/ADA290387`.
15. Martin Gardner. Mathematical games: Tasks you cannot help finishing no matter how hard you try to block finishing them, *Scientific American*, 24(2):12–21. Reprinted in Martin Gardner, *The Last Recreations*, pp. 27–43, Springer Verlag, 1998.
16. Jürgen Giesl, Thomas Arts and Enno Ohlebusch. Modular termination proofs for rewriting using dependency pairs. *J. of Symbolic Computation*, 34:21–58, 2002.
17. Reuben L. Goodstein. On the restricted ordinal theorem, *J. Symbolic Logic*, 9:33–41, 1944.
18. David Gries. Is sometimes ever better than alway? *ACM Transactions on Programming Languages and Systems*, 1(2):258–265, 1979. `http://doi.acm.org/10.1145/357073.357080`.
19. Jaap M. Hemelrijk. Caeretan Hydriae. *American Journal of Archaeology*, 89(4): 701–703, 1985.
20. Nao Hirokawa and Aart Middeldorp. Dependency pairs revisited. *Proceedings of the 15th International Conference on Rewriting Techniques and Applications*, Lecture Notes in Computer Science, vol. 3091, pp. 249–268, Springer Verlag, 2004.
21. Bernard R. Hodgson. Herculean or Sisyphean tasks. *Newsletter of the European Mathematical Society*, No. 51, 2004.
22. Thomas J. Jech. *Set Theory*. Springer Verlag, 2002.

23. Jean-Pierre Jouannaud. Proof and computation. In H. Schwichtenberg, ed., NATO series F: *Computer and Systems Sciences*, vol. 139, pp. 173–218, Springer Verlag, 1995. `http://rewriting.loria.fr/documents/rpac.ps.gz`.
24. Laurie Kirby and Jeff Paris. Accessible independence results for Peano arithmetic. *Bull. London Mathematical Society*, 4:285–293, 1982.
25. Ingo Lepper. Simply terminating rewrite systems with long derivations. *Arch. Math. Logic*, 43:1–18, 2004.
26. Pierre Lescanne, ed. Rewriting mailing list. `https://listes.ens-lyon.fr/wws/arc/rewriting`, February 19, 2004.
27. Fabrizio Luccio and Linda Pagli. Death of a monster. *SIGACT News*, 31(4):130–133, 2000. `http://doi.acm.org/10.1145/369836.369904`.
28. Claude Marché and Hans Zantema. The termination competition. Franz Baader, ed., *Proceedings of the 18th International Conference on Rewriting Techniques and Applications*, Lecture Notes in Computer Science, Springer Verlag, 2007.
29. Georg Moser and Andreas Weiermann. Relating derivation lengths with the slow-growing hierarchy directly. *Proceedings of the 14th International Conference on Rewriting Techniques and Applications*, Lecture Notes in Computer Science, vol. 2706, pp. 296–310, Springer Verlag, 2003.
30. Jeff Paris and Leo Harrington. A mathematical incompleteness in Peano arithmetic. In *Handbook for Mathematical Logic*, J. Barwise, ed., North-Holland, 1977.
31. Penn State College of Information Sciences and Technology. CiteSeer Scientific Literature Digital Library. `http://citeseer.ist.psu.edu`.
32. Gaisi Takeuti. *Proof Theory*. North-Holland, Amsterdam, 2nd edition, 1987.
33. Terese (Marc Bezem, Jan Willem Klop and Roel de Vrijer, eds.). *Term Rewriting Systems*, volume 55 of *Cambridge Tracks in Theoretical Computer Science*. Cambridge University Press, 2003.
34. Hélène Touzet. Encoding the Hydra battle as a rewrite system. *Proceedings of the the 23rd International Symposium on Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science, vol. 1450, pp. 267–276, Springer Verlag, 1998.
35. Alan M. Turing. Checking a large routine. *Report of a Conference on High Speed Automatic Calculating Machines*, Univ. Math. Lab., Cambridge, pp. 67-69, 1949. Reprinted in F. L. Morris and C. B. Jones. An early program proof by Alan Turing. *Annals of the History of Computing*, 6:139–143, 1984. `http://www.turingarchive.org/browse.php/B/8`.
36. Stan S. Wainer. Ordinal recursion, and a refinement of the extended Grzegorezyk hierarchy. *J. Symbolic Logic*, 37:281–292, 1972.