

A Characterisation of Multiply Recursive Functions with Higman’s Lemma

Hélène Touzet

LIFL—Université Lille 1, 59 655 Villeneuve d’Ascq, France
E-mail: touzet@lil.fr

Received September 15, 1999; revised February 6, 2001; published online August 9, 2002

We prove that string rewriting systems which reduce by Higman’s lemma exhaust the multiply recursive functions. This result provides a full characterisation of the expressiveness of Higman’s lemma when applied to rewriting theory. The underlying argument of our construction is to connect the order type and the derivation length via the Hardy hierarchy. © 2002 Elsevier Science (USA)

1. INTRODUCTION

1.1. Higman’s Lemma

Recall the statement of Higman’s lemma for strings. Given an alphabet \mathcal{A} , define the *division ordering* Θ as the least preorder on the set of finite strings \mathcal{A}^* satisfying the following properties

- *subterm property*: $\forall a \in \mathcal{A}, \forall u \in \mathcal{A}^*, u \Delta au$,
- *strict monotonicity*: $\forall u, v \in \mathcal{A}^*, \forall a \in \mathcal{A}, u \Delta v \Rightarrow au \Delta av$.

THEOREM 1.1 (Higman [7]). *For any finite alphabet \mathcal{A} , (\mathcal{A}^*, Θ) is a well-partial-ordering.*

A well-partial-ordering is a well-founded ordering with no infinite antichain. In other words, every ordering extending Θ is still well founded. So Higman’s lemma provides a syntactic criterion for the definition of well-founded orderings on strings. Let us mention the Knuth–Bendix ordering, the recursive path ordering, and the polynomial orderings. What concerns us is the expressiveness of string rewriting systems (SRS). Given a Noetherian finite SRS \mathcal{R} on an alphabet \mathcal{A} , define the *derivation length function* $Dl_{\mathcal{R}}$ by

$$\begin{aligned} dl_{\mathcal{R}}: \mathcal{A}^* &\rightarrow \mathbb{N} \\ w &\mapsto \max\{dl_{\mathcal{R}}(u), w \rightarrow_{\mathcal{R}} u\} + 1 \\ Dl_{\mathcal{R}}: \mathbb{N} &\rightarrow \mathbb{N} \\ m &\mapsto \max\{n \in \mathbb{N}, \exists w \in \mathcal{A}^*, dl_{\mathcal{R}}(w) = n \wedge |w| \leq m\}, \end{aligned}$$

where $|w|$ is the size of the string w . The expressiveness of the main termination orderings was extensively studied and we know that most ensure primitive recursive derivation lengths on strings (see [8] for the Knuth–Bendix and polynomial orderings and [9] for the recursive path ordering). The purpose of this paper is to investigate the derivation length of the whole class of string rewriting systems reducing by Higman’s lemma. Extending the result of [13], we establish that existing termination proof techniques do not reach the full strength of Higman’s lemma: One can go far beyond primitive recursiveness and exhaust the class of multiply recursive functions.

1.2. Multiply Recursive Functions and the Hardy Hierarchy

Multiply recursive functions are traditionally defined by closure under the schemes of k -recursion (see Peter [11] for instance). Grzegorzcyk, Wainer, and others teach us that classes of functions may also be described by hierarchies of functions indexed by ordinals. We adopt this alternative point of view here and introduce the class of multiply recursive functions by means of the Hardy hierarchy. Let $\mathcal{CNF}(\varepsilon_0)$ be the set of notations in Cantor normal form for ordinals below ε_0 . A canonical assignment

of fundamental sequences for limit ordinals in $\mathcal{CNF}(\varepsilon_0)$ is defined recursively as

$$\begin{aligned} \omega_n &= n \\ (\alpha + \lambda)_n &= \alpha + \lambda_n \\ (\omega^{\beta+1})_n &= \omega^\beta n \\ (\omega^\lambda)_n &= \omega^{\lambda_n}, \end{aligned}$$

where β is in $\mathcal{CNF}(\varepsilon_0)$ and λ is a limit ordinal in $\mathcal{CNF}(\varepsilon_0)$. With the fundamental sequences, one can define the family of *predecessor functions*.¹ For each $n \in \mathbb{N}$, $P_n : \mathcal{CNF}(\varepsilon_0) \rightarrow \mathcal{CNF}(\varepsilon_0)$ is

$$\begin{aligned} P_n(0) &= 0 \\ P_n(\alpha + 1) &= \alpha \\ P_n(\lambda) &= \lambda_n, \quad \text{if } \lambda \text{ is a limit ordinal.} \end{aligned}$$

For each ordinal α of $\mathcal{CNF}(\varepsilon_0)$, the *Hardy function* \mathcal{H}_α is now defined as follows.

$$\begin{aligned} \mathcal{H}_0: \mathbb{N} &\rightarrow \mathbb{N} \\ n &\mapsto 0 \end{aligned}$$

and for $\alpha > 0$

$$\begin{aligned} \mathcal{H}_\alpha: \mathbb{N} &\rightarrow \mathbb{N} \\ n &\mapsto \mathcal{H}_{P_n(\alpha)}(n + 1) + 1. \end{aligned}$$

The class of *multiply recursive functions* is exactly described by the family of Hardy functions indexed by ordinals of ω^{ω^ω} (Robbin [12]).

Why should we give a preponderant role to the Hardy hierarchy? Consider the maximal order type of the division ordering Θ . De Jongh and Parikh established in [3] that for any finite nonempty alphabet \mathcal{A} , the maximal order type of (\mathcal{A}^*, Θ) is $\omega^{|\mathcal{A}|-1}$. Cichon and Tahhan Bittar took advantage of this result and produced a measure for sequences compatible with Θ using the Hardy hierarchy indexed by the maximal order type of the division ordering.

THEOREM 1.2 (Cichon and Tahhan Bittar [2]). *Let \mathcal{A} be a finite alphabet and $k \in \mathbb{N}$. For each string u in \mathcal{A}^* , $|u|$ denotes the size of u . There is a function $\phi : \mathbb{N} \rightarrow \mathbb{N}$ such that for any sequence $(u_i)_{i \in \mathbb{N}}$ of \mathcal{A}^* satisfying*

- $\forall i, j \in \mathbb{N}, i < j \Rightarrow \neg(u_i \Theta u_j),$
- $\forall i \in \mathbb{N}, |u_i| \leq |u_0| + k \times i,$

the length of $(u_i)_{i \in \mathbb{N}}$ is bounded by $\phi(|u_0|)$. Moreover ϕ is an elementary function in $\mathcal{H}_{\omega^{|\mathcal{A}|}}$.

This theorem provides an upper bound for rewrite derivations: Each finite string rewriting system reducing by Higman's lemma has a multiply recursive derivation length.

We investigate here the intriguing role of the Hardy hierarchy and show that it is possible to encode Hardy functions indexed by ordinals of ω^{ω^ω} by finite string rewriting systems which are compatible with the division ordering of Higman's lemma. This construction shows that Cichon and Tahhan Bittar's upper bound is essentially optimal. At a logical level it confirms the fact that the Hardy hierarchy is the right tool for connecting derivation length and order type. The proof goes as follows. The Hardy hierarchy enjoys an intuitive geometrical description: For each ordinal α and all integers n , consider the decreasing sequence of ordinals $(\alpha_i)_{i \in \mathbb{N}}$ given by

$$(\star) \quad \begin{aligned} \alpha_0 &= \alpha, \\ \alpha_{i+1} &= P_{n+i}(\alpha_i). \end{aligned}$$

¹ For the limit case, the predecessor function is sometimes defined as $P_n(\lambda) = P_n(\lambda_n)$. This does not affect the complexity of the Hardy hierarchy.

The sequence stops when it reaches 0. We call it a *Hardy sequence*. $\mathcal{H}_\alpha(n)$ is simply the length of the Hardy sequence generated by α and n . We use this representation and encode Hardy sequences for ordinals below ω^{ω^ω} by rewrite systems. This idea is already present in [14], where it leads to a new lower bound of the complexity of simplifying *term* rewriting systems. Here we use strings instead of terms. First, we need to produce a specific notation system for ordinals below ω^{ω^ω} based on strings. For that we choose the recursive path ordering of Dershowitz (Section 3). We are then able to construct the string rewriting systems and their proof of termination by Higman's lemma (Section 4). We even establish *total termination*.

2. REWRITING THEORY AND TOTAL TERMINATION

We do not recall fundamental notions on rewrite systems and termination (see [5] for instance). Let \mathcal{A} be an alphabet and $<$ an ordering on \mathcal{A}^* . We say that

$<$ is *strictly monotone* if $\forall u, v \in \mathcal{A}^*, (u < v \Rightarrow \forall a \in \mathcal{A} \text{ } au < av)$.

$<$ is *monotone* if $\forall u, v \in \mathcal{A}^*, (u \leq v \Rightarrow \forall a \in \mathcal{A}, au \leq av)$, where \leq is the reflexive closure of $<$.

$<$ is *stable* if $\forall u, v \in \mathcal{A}^*, (u < v \Rightarrow \forall a, b \in \mathcal{A}, aub \leq avb)$,

$<$ has the *subterm property* if $\forall u \in \mathcal{A}^*, \forall a \in \mathcal{A}, u < au$.

A rewrite system \mathcal{R} is *compatible* with the ordering $<$ if, and only if,

$$\forall u, v \in \mathcal{A}^*, \quad u \rightarrow_{\mathcal{R}} v \Rightarrow v < u.$$

We now come to the definition of total termination. The original concept is due to Ferreira and Zanema [6].

DEFINITION 2.1 (Total termination). Let \mathcal{A} be an alphabet. A *total termination* ordering on \mathcal{A}^* is a strictly monotone well-order. A rewrite system \mathcal{R} of \mathcal{A}^* is *totally terminating* if it is compatible with a total termination ordering.

Total termination on a finite alphabet implies the subterm property (see [4] or [6]). It follows that any totally terminating rewrite system is compatible with the division ordering of Higman's theorem. We now give another characterisation of total termination, which requires only monotonicity, instead of strict monotonicity. This result is useful in Section 4.

PROPOSITION 2.1. Let \mathcal{A} be a finite alphabet and let \mathcal{R} be a rewrite system on \mathcal{A}^* . \mathcal{R} is *totally terminating* if and only if there exists a total preordering \leq on \mathcal{A}^* such that

- (i) for each $l \rightarrow r$ in \mathcal{R} , for each $u \in \mathcal{A}^*$, $lu > ru$,
- (ii) $<$ has the subterm property,
- (iii) $<$ is monotone.

Proof. One direction of the proof is obvious. For the other direction, if \leq is a preordering, not an ordering, we can make it antireflexive by combining it lexicographically with any total ordering on \mathcal{A}^* . Properties (i), (ii), and (iii) remain unchanged. We now assume that \leq is a total ordering. Let $\text{mul}(\mathcal{A}^*)$ denote the set of finite multisets on \mathcal{A}^* , $\text{mul}(<)$ the multiset extension of $<$ on $\text{mul}(\mathcal{A}^*)$, and \cup the union of multisets. For each string u in \mathcal{A}^* , define $\mathcal{M}(u)$ as the multiset containing u and its suffixes,

$$\begin{aligned} \mathcal{M}(\varepsilon) &= \emptyset, \\ \mathcal{M}(au) &= \{au\} \cup \mathcal{M}(u), \end{aligned}$$

and define $<'$ as

$$u <' v \Leftrightarrow \mathcal{M}(u) \text{mul}(<) \mathcal{M}(v).$$

We claim that $<'$ is a total termination ordering for \mathcal{R} . First, $<'$ is strictly monotone: Let $u, v \in \mathcal{A}^*$ such that $u <' v$ and let $a \in \mathcal{A}$. We have

$$\begin{aligned} \mathcal{M}(au) &= \{au\} \cup \mathcal{M}(u), \\ \mathcal{M}(av) &= \{av\} \cup \mathcal{M}(v). \end{aligned}$$

By hypothesis, we have $\mathcal{M}(u) \text{mul}(<) \mathcal{M}(v)$. So it suffices to show that $au \leq av$. Suppose $u > v$. By hypothesis (ii), this would imply $\{u\} \text{mul}(>) \mathcal{M}(v)$; thus $\mathcal{M}(u) \text{mul}(>) \mathcal{M}(v)$, which contradicts the hypothesis $\mathcal{M}(u) \text{mul}(<) \mathcal{M}(v)$. So $u \leq v$, which with (iii) ensures $au \leq av$. Thus $au <' av$. Second, $<'$ is well founded, since it is strictly monotone and enjoys the subterm property. Finally, $<'$ is compatible with \mathcal{R} : Let $l \rightarrow r$ in \mathcal{R} and $u \in \mathcal{A}^*$. By (i), $lu > ru$, which with (ii) implies $\mathcal{M}(lu) \text{mul}(>) \mathcal{M}(ru)$. Hence $lu >' ru$. This completes the proof. ■

Remark. In Definition 2.1, the notion of total termination on strings coincides with the usual definition on terms: It uses only (left-) monotonicity and not stability (left and right monotonicity). So our total termination orderings are not total division orderings, such as those studied in [10]. There is a slight difference between those two families of orderings: For instance the string rewriting system

$$\begin{cases} ff \rightarrow gf \\ gg \rightarrow fg \end{cases}$$

is totally terminating, but it does not reduce under any total division ordering.

3. ORDINAL NOTATIONS FOR $\omega^{\omega^{\omega}}$ WITH STRINGS

The core of our construction is to simulate decreasing sequences of ordinals of $\omega^{\omega^{\omega}}$ by sequences of strings. For that, we are now going to introduce an ordinal notation system that is based on the recursive path ordering on strings.

DEFINITION 3.1 (Recursive path ordering [4]). Let \mathcal{A} be an alphabet equipped with the precedence $<$. The *recursive path ordering* $<_{rpo}$ is the least stable ordering which satisfies

- if $u \Delta v$, then $u <_{rpo} v$,
- if $u <_{rpo} bv$ and $a < b$, then $au <_{rpo} bv$.

PROPOSITION 3.1 [4]. *If $(\mathcal{A}, <)$ is a well-order, then $(\mathcal{A}^*, <_{rpo})$ is a total termination ordering.*

In the remainder of the paper, we choose $\mathcal{A} = \{a_i; i \in \mathbb{N}\}$ with the well-ordered precedence induced by the transitive closure of $a_i < a_{i+1}$. It is routine to prove that the order type of $<_{rpo}$ on \mathcal{A}^* is $\omega^{\omega^{\omega}}$, the maximal order type of the division ordering. It means that there exists an isomorphism \mathcal{O} of $\omega^{\omega^{\omega}} \rightarrow (\mathcal{A}^*, <_{rpo})$ such that each ordinal of $\omega^{\omega^{\omega}}$ may be denoted in a unique and nonambiguous way by a string of \mathcal{A}^* . The purpose of the remainder of the section is to make the construction of \mathcal{O} explicit.

PROPOSITION 3.2. *For each limit ordinal α in $\omega^{\omega^{\omega}}$, there are unique $i \in \mathbb{N}$, $\beta, \gamma \in \omega^{\omega^{\omega}}$ such that*

$$\alpha = \gamma + \omega^{\omega^i} \beta$$

and satisfying

- (i) $0 < \beta < \omega^{\omega^{i+1}}$,
- (ii) $\forall 0 < \mu < \omega^{\omega^{i+1}}, \forall \delta \in \omega^{\omega^{\omega}} \gamma \neq \delta + \mu$.

Proof. Let $\omega^{\alpha_1} + \dots + \omega^{\alpha_n}$ be the Cantor normal form of α . Since α is a limit ordinal of $\omega^{\omega^{\omega}}$, we have $0 < \alpha_n \leq \dots \leq \alpha_1 < \omega^{\omega}$. Let $i \in \mathbb{N}$ such that $\omega^i \leq \alpha_n < \omega^{i+1}$ and let j be the smallest index such

that $\omega^i \leq \alpha_j < \omega^{i+1}$. There are $\delta_j, \dots, \delta_n$ in ω^{i+1} such that

$$\begin{aligned} \alpha_j &= \omega^i + \delta_j, \\ &\vdots \\ \alpha_n &= \omega^i + \delta_n. \end{aligned}$$

If we set $\gamma = \omega^{\alpha_1} + \dots + \omega^{\alpha_{j-1}}$ (γ is possibly 0) and $\beta = \omega^{\delta_j} + \dots + \omega^{\delta_n}$, then i, β, γ satisfy conditions (i), (ii) and $\alpha = \gamma + \omega^i \beta$.

We now prove that this decomposition is unique. Let β, γ, i , and i' satisfy conditions (i), (ii) such that $\gamma + \omega^i \beta = \gamma' + \omega^{i'} \beta'$. Consider the Cantor normal form of β and β' : $\beta = \omega^{\beta_1} + \dots + \omega^{\beta_n}$ and $\beta' = \omega^{\beta'_1} + \dots + \omega^{\beta'_m}$. If we require that γ and γ' are in Cantor normal form too, conditions (ii), (iii) guarantee that the notations $\gamma + \omega^{\omega^j + \beta_1} + \dots + \omega^{\omega^j + \beta_n}$ and $\gamma' + \omega^{\omega^{j'} + \beta'_1} + \dots + \omega^{\omega^{j'} + \beta'_m}$ are two Cantor normal forms of the same ordinal. They are identical. It implies $i = i'$. Suppose now that $\gamma < \gamma'$ (for instance). It would follow that $\gamma' = \gamma + \omega^{\omega^j + \beta_1} + \dots + \omega^{\omega^j + \beta_k}$ for some $k \leq n$, which contradicts (ii). So $\gamma = \gamma'$ and then $n = m, \beta_1 = \beta'_1, \dots, \beta_n = \beta'_n$. ■

In Proposition 3.2, we require that $\beta > 0$. In this case, we define $-1 + \beta$ as the unique ordinal such that $1 + (-1 + \beta) = \beta$. Recall that for infinite ordinals $1 + \beta = \beta$, and for finite ordinals $1 + \beta = \beta + 1$. So $-1 + \beta = \beta$ when β is infinite and $-1 + \beta = \beta - 1$ when β is finite and nonempty. This leads to the definition of the notation system \mathcal{O} .

DEFINITION 3.2 (The notation system \mathcal{O}).

$$\begin{aligned} \mathcal{O}: \quad \omega^{\omega^\omega} &\rightarrow \mathcal{A}^* \\ 0 &\mapsto \varepsilon \\ \beta + 1 &\mapsto a_0 \mathcal{O}(\beta) \\ \gamma + \omega^i \beta &\mapsto a_{i+1} \mathcal{O}(-1 + \beta) \mathcal{O}(\gamma) \end{aligned}$$

EXAMPLE 3.1. $\mathcal{O}(n) = a_0^n, \mathcal{O}(\omega) = a_1, \mathcal{O}(\omega + n) = a_0^n a_1, \mathcal{O}(\omega + \omega) = a_1 a_0, \mathcal{O}(\omega^2) = a_1 a_1, \mathcal{O}(\omega^\omega) = a_2, \mathcal{O}((\omega^\omega)_n) = a_i^n$.

PROPOSITION 3.3. \mathcal{O} is an isomorphism of $(\omega^{\omega^\omega}, <) \rightarrow (\mathcal{A}^*, <_{rpo})$.

Proof. We first prove that for all $\alpha, \beta \in \omega^{\omega^\omega}$, if $\alpha < \beta$ then $\mathcal{O}(\alpha) <_{rpo} \mathcal{O}(\beta)$. The ordinal ordering $<$ is the transitive closure of the schemes

$$\begin{aligned} \beta &< \beta + 1, \\ \forall n \in \mathbb{N} \quad \gamma + \omega^i \beta + (\omega^\omega)_n &< \gamma + \omega^i (\beta + 1), \\ \forall n \in \mathbb{N} \quad \gamma + \omega^i \beta_n &< \gamma + \omega^i \beta \quad (\beta \text{ limit}). \end{aligned}$$

On strings, it corresponds to the three following inequalities:

$$\begin{aligned} u &<_{rpo} a_0 u, \\ \forall n \in \mathbb{N} \quad a_i^n a_{i+1} \mathcal{O}(\beta) \mathcal{O}(\gamma) &<_{rpo} a_{i+1} a_0 \mathcal{O}(\beta) \mathcal{O}(\gamma), \\ \forall n \in \mathbb{N} \quad a_{i+1} \mathcal{O}(\beta_n) \mathcal{O}(\gamma) &<_{rpo} a_{i+1} \mathcal{O}(\beta) \mathcal{O}(\gamma). \end{aligned}$$

The proof is direct, using the definition of $<_{rpo}$ with an easy induction on β . As a consequence, \mathcal{O} is an injective morphism. It remains to show that \mathcal{O} is surjective. Let $u \in \mathcal{A}^*$. We construct by induction on the length of u an ordinal α such that $\mathcal{O}(\alpha) = u$. If $u = a_0 v$ for some $v \in \mathcal{A}^*$, then the induction hypothesis gives us an ordinal β such that $\mathcal{O}(\beta) = v$. Set $\alpha = \beta + 1$. The definition of \mathcal{O} ensures $\mathcal{O}(\alpha) = a_0 v$. If $u = a_{i+1} v$ for some $v \in \mathcal{A}^*$ and some $i \in \mathbb{N}$, we have to consider two subcases. When $v \in \{a_0, \dots, a_{i+1}\}^*$, let β be such that $\mathcal{O}(-1 + \beta) = v$. In this case, $\beta < \omega^{\omega^{i+1}}$. So $\mathcal{O}(\omega^i \beta) = a_{i+1} v$. Otherwise, there exist $b \in \{a_{i+2}, \dots\}, v_1 \in \{a_0, \dots, a_{i+1}\}^*$ and $v_2 \in \mathcal{A}^*$ such that $v = v_1 b v_2$. Let β and γ be such that $\mathcal{O}(-1 + \beta) = v_1$ and $\mathcal{O}(\gamma) = b v_2$. We have $\mathcal{O}(\gamma + \omega^i \beta) = a_{i+1} v_1 b v_2 = a_{i+1} v$. ■

From now on, we shall always assume that a string built up from the alphabet $\mathcal{A} = \{a_0, \dots, a_i, \dots\}$ is a notation for an ordinal of ω^{ω^ω} . We express the predecessor functions in this notation system.

PROPOSITION 3.4. *For each $u \in \mathcal{A}^*$, for all $i, j, n \in \mathbb{N}$*

- (i) $P_n(a_0u) = u$,
- (ii) $P_n(a_{i+1}a_0u) = a_i^n a_{i+1}u$,
- (iii) $P_n(a_{i+1}a_ju) = a_{i+1}P_n(a_ju)$, when $0 < j \leq i + 1$,
- (iv) $P_n(a_{i+1}u) = a_i^n u$, otherwise.

Proof. Given an ordinal α of ω^{ω^ω} , we distinguish four main cases for the computation of $P_n(\alpha)$:

Case 1. $\alpha = \delta + 1$. Then $P_n(\alpha) = \delta$.

In all remaining cases, α is a limit ordinal. Let i, β, γ satisfy conditions (i), (ii) of Proposition 3.2 such that $\alpha = \gamma + \omega^{\omega^i} \beta$.

Case 2. $\alpha = \gamma + \omega^{\omega^i} (\beta' + 1)$, $i = 0$, $\beta' \neq 0$. Then $P_n(\alpha) = \gamma + \omega^{\omega^i} \beta' + (\omega^{\omega^i})_n$.

Case 3. $\alpha = \gamma + \omega^{\omega^i} \beta$, β limit. Then $P_n(\alpha) = \gamma + \omega^{\omega^i} \beta_n$.

Case 4. $\alpha = \gamma + \omega^{\omega^i}$. Then $P_n(\alpha) = \gamma + (\omega^{\omega^i})_n$.

We apply the notation \mathcal{O} to α and $P_n(\alpha)$. The proof is by induction on the length of u .

Case 1. $\mathcal{O}(\alpha) = a_0 \mathcal{O}(\delta)$ and $\mathcal{O}(P_n(\alpha)) = \mathcal{O}(\delta)$.

Case 2. $\mathcal{O}(\alpha) = a_{i+1} a_0 \mathcal{O}(\beta') \mathcal{O}(\gamma)$ and $\mathcal{O}(P_n(\alpha)) = a_i^n a_{i+1} \mathcal{O}(\beta') \mathcal{O}(\gamma)$.

Case 3. $\mathcal{O}(\alpha) = a_{i+1} \mathcal{O}(\beta) \mathcal{O}(\gamma)$ and $\mathcal{O}(P_n(\alpha)) = a_{i+1} \mathcal{O}(\beta_n) \mathcal{O}(\gamma)$. By induction hypothesis, note that for all $j \in \mathbb{N}$, $u \in \{a_0, \dots, a_j\}^+$ and $v \in \mathcal{A}^*$ $P_n(ua_{j+1}v) = P_n(u)a_{j+1}v$. So $\mathcal{O}(P_n(\alpha)) = a_{i+1} P_n(\mathcal{O}(\beta) \mathcal{O}(\gamma))$.

Case 4. $\mathcal{O}(\alpha) = a_{i+1} \mathcal{O}(\gamma)$ and $\mathcal{O}(P_n(\alpha)) = a_i^n \mathcal{O}(\gamma)$.

Thus, (i) comes from case 1, (ii) from case 2, (iii) from case 3, and (iv) from case 4. ■

4. ENCODING MULTIPLY RECURSIVE FUNCTIONS BY TOTALLY TERMINATING SRS'S

4.1. Construction of the SRS

In this section, we build a family of string rewriting systems that simulate the Hardy hierarchy using the notation system based on the recursive path ordering of Section 3. Let us illustrate our approach by a simple example. We consider the following Hardy sequence \mathcal{H} .

$$(\omega^2, 2); (\omega 2, 3); (\omega + 3, 4); (\omega + 2, 5); (\omega + 1, 6); (\omega, 7); (7, 8)$$

In the string notation system, \mathcal{H} is written

$$(a_1 a_1, 2); (a_1 a_0, 3); (a_0^3 a_1, 4); (a_0 a_0 a_1, 5); (a_0 a_1, 6); (a_1, 7); (a_0^7, 8).$$

It remains to transform each couple (u, n) of the sequence into a single string. For that, we introduce a new symbol $|$ and denote the integer n in unary notation: $|^n$. Then we concatenate this natural number to the ordinal notation. So a pair of the form (u, n) is denoted by the string $|^n u$. For \mathcal{H} , it gives

$$||a_1 a_1; |||a_1 a_0; ||||a_0^3 a_1; |||||a_0 a_0 a_1; |||||a_0 a_1; |||||a_1; |||||a_0^7.$$

In each step of the sequence, the calculation will involve two phases: Counting the number of $|$, n and then computing the n th predecessor of u according to the four cases of Proposition 3.4. To deal with technical details in the computation, we need extra symbols: \circ and \bullet for the counting phase, and k_i for the predecessor phase. Each step (u, n) of the Hardy sequence is finally encoded by the string $\bullet|^n u$.

It is hopeless to try to simulate the whole class of multiply recursive functions by a single totally terminating finite string rewriting system. We define a family \mathcal{R}_i , $i \in \mathbb{N}$, such that \mathcal{R}_i exhausts the Hardy hierarchy on ω^{ω^i} . For each $u \in \{a_0, \dots, a_i\}^*$, the system \mathcal{R}_i should then allow us to derive

$$\bullet |^n u \xrightarrow{+}_{\mathcal{R}_i} \bullet |^{n+1} P_n(u).$$

We set

$$\mathcal{R}_0 \left\{ \begin{array}{ll} a_0 \rightarrow \circ & (0, 1) \\ \circ \rightarrow \bullet | & (0, 2) \\ \bullet | \rightarrow | \bullet \bullet & (0, 3) \\ | \circ \rightarrow \circ | & (0, 4) \\ \bullet \rightarrow \varepsilon & (0, 5) \end{array} \right.$$

and

$$\mathcal{R}_{i+1} = \mathcal{R}_i \cup \left\{ \begin{array}{ll} \bullet a_{i+1} a_0 \rightarrow k_{i+1} a_{i+1} & (i+1, 1) \\ \bullet a_{i+1} \rightarrow a_{i+1} \bullet & (i+1, 2) \\ \bullet k_{i+1} \rightarrow k_{i+1} a_i & (i+1, 3) \\ a_{i+1} \circ \rightarrow \circ a_{i+1} & (i+1, 4) \\ k_{i+1} \rightarrow \circ & (i+1, 5) \\ \bullet a_{i+1} \rightarrow k_{i+1} & (i+1, 6). \end{array} \right.$$

Figure 1 shows an example of derivation corresponding to the Hardy sequence \mathcal{H} , starting from $(\omega^2, 2)$. We now establish that our family of rewrite systems is correct for the encoding of all Hardy sequences of ω^{ω^ω} .

PROPOSITION 4.1. *Given $u \in \{a_0, \dots, a_{i+1}\}^*$ and $n \geq 1$, $\bullet |^n u \xrightarrow{+}_{\mathcal{R}_{i+1}} \bullet |^{n+1} P_n(u)$.*

Proof. Define $(\spadesuit) \bullet |^{n+1} u \xrightarrow{+} \circ P_n(u)$. (\spadesuit) implies the desired result.

$$\begin{aligned} \bullet |^n u &\xrightarrow{+} |^n \bullet^{2^n} u && (0, 3)^+ \\ &\xrightarrow{*} |^n \bullet^{n+1} u && (0, 5)^* \\ &\xrightarrow{+} |^n \circ P_n(u) && (\spadesuit) \\ &\xrightarrow{+} \circ |^n P_n(u) && (0, 4)^+ \\ &\xrightarrow{+} \bullet |^{n+1} P_n(u) && (0, 2) \end{aligned}$$

(\spadesuit) is now proved by induction on u . We consider the four cases introduced in Proposition 3.4.

Case 1.

$$\begin{aligned} \bullet^{n+1} a_0 v &\rightarrow a_0 v && (0, 5)^* \\ &\rightarrow \circ v && (0, 1) \end{aligned}$$

Case 2.

$$\begin{aligned} \bullet^{n+1} a_{i+1} a_0 v &\rightarrow \bullet^n k_{i+1} a_{i+1} v && (i+1, 1) \\ &\xrightarrow{*} k_{i+1} a_i^n a_{i+1} v && (i+1, 3)^* \\ &\rightarrow \circ a_i^n a_{i+1} v && (i+1, 5) \end{aligned}$$

Case 3.

$$\begin{aligned} \bullet^{n+1} a_{i+1} v &\xrightarrow{+} a_{i+1} \bullet^{n+1} v && (i+1, 2)^* \\ &\xrightarrow{*} a_{i+1} \circ P_n(v) && (\text{induction hypothesis}) \\ &\rightarrow \circ a_{i+1} P_n(v) && (i+1, 4) \end{aligned}$$

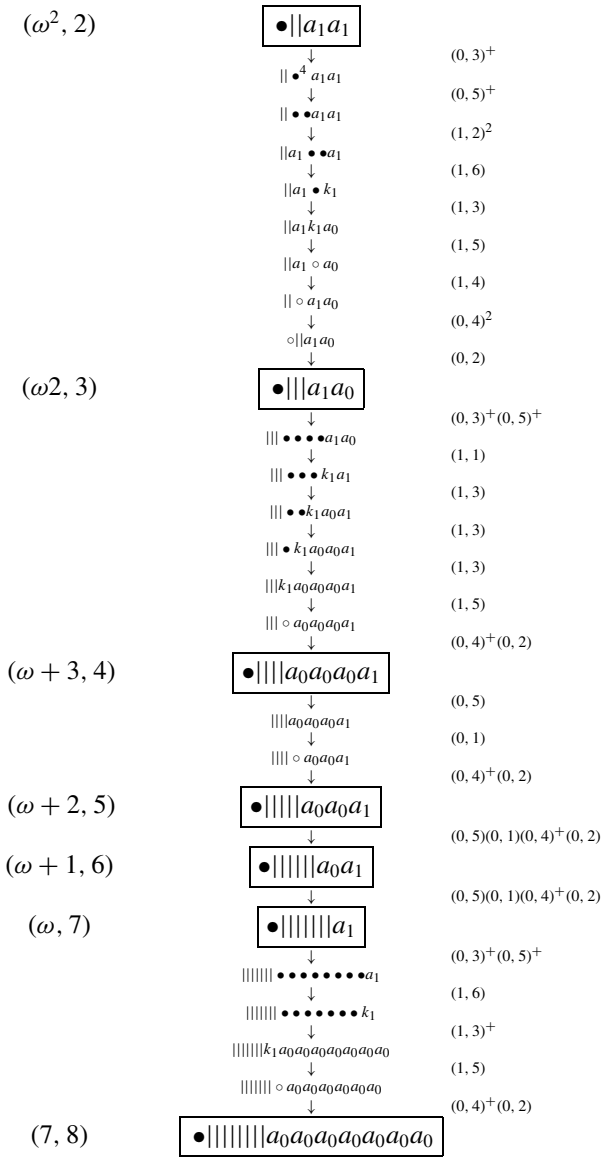


FIG. 1. Rewrite derivation for the Hardy sequence \mathcal{H} .

Case 4.

$$\begin{aligned}
 \bullet^{n+1} a_{i+1} v &\rightarrow \bullet^n k_{i+1} v && (i+1, 6) \\
 &\rightarrow k_{i+1} a_i^n v && (i+1, 3)^* \\
 &\rightarrow \circ a_i^n v && (i+1, 5)
 \end{aligned}$$

■

COROLLARY 4.1. For each multiply recursive function f , there exists i in \mathbb{N} such that f is eventually dominated by $Dl_{\mathcal{R}_i}$, the derivation length of \mathcal{R}_i .

Proof. Proposition 4.1 implies that for each i in \mathbb{N} , $Dl_{\mathcal{R}_{i+1}}$ eventually dominates the Hardy function indexed by ω^{ω^i} . ■

4.2. For Each $i \in \mathbb{N}$, \mathcal{R}_i is Totally Terminating

We now come to the final argument of our construction and show that the string rewriting systems \mathcal{R}_i are totally terminating. For each ordinal u , note that the string $\bullet|^n u$ is greater than $\bullet|^n P_n(u)$ wrt the recursive path ordering with the precedence $|\prec a_0$. This property may be used to establish termination, but unfortunately not total termination. Nevertheless, the recursive path ordering will indirectly be useful for our proof of total termination. The starting point of the proof is Proposition 2.1: We define for \mathcal{R}_i a monotone preordering which enjoys the subterm property. For that, we rely on the semantics of the symbols of \mathcal{A} : Strings built up from a_0, \dots, a_i, \dots may simply be ordered by the recursive path ordering. For the symbol k_i , define the function ψ_i by

$$\psi_{i+1}: u \mapsto \sup\{a_i^n u; n \in \mathbb{N}\}.$$

(The supremum is wrt \prec_{rpo} .) We have the following properties.

LEMMA 4.1. For each u in \mathcal{A}^* , for each i in \mathbb{N}

- (i) $\psi_{i+1}(u) = \psi_{i+1}(a_i u)$,
- (ii) $a_{i+1} a_0 u \succeq_{rpo} \psi_{i+1}(a_{i+1} u)$,
- (iii) $a_{i+1} u \succeq_{rpo} \psi_{i+1}(u)$,
- (iv) $\psi_i(u) \succ_{rpo} u$,
- (v) ψ_i is an increasing function.

Proof. (i) is by definition of ψ_i and (ii), (iii), (iv), (v) are easy consequences of the definition of \leq_{rpo} . ■

For the symbols \bullet , \circ , and $|$, consider the subsystem

$$\mathcal{S} \begin{cases} \circ \rightarrow \bullet | \\ \bullet | \rightarrow | \bullet \bullet \\ | \circ \rightarrow \circ | \\ \bullet \rightarrow \varepsilon \\ \bullet a_{i+1} \rightarrow a_{i+1} \bullet \\ a_{i+1} \circ \rightarrow \circ a_{i+1}. \end{cases}$$

LEMMA 4.2. There exists a total termination ordering $\prec_{\mathcal{S}}$ for \mathcal{S} .

Proof. We give an interpretation \mathcal{I} on \mathbb{N}^3 for the rules of \mathcal{S} .

$$\begin{aligned} \mathcal{I}(\circ)(n, m, p) &= (2n + 4, m, p) \\ \mathcal{I}(|)(n, m, p) &= (2n + 1, m, p) \\ \mathcal{I}(\bullet)(n, m, p) &= (n, n + m, 2p + 2) \\ \mathcal{I}(a_i)(n, m, p) &= (n, n + m, p + 1) \end{aligned}$$

Define $u \prec_{\mathcal{S}} v$ by $\mathcal{I}(u) < \mathcal{I}(v)$, where 3-uplets are compared in the usual left to right lexicographic ordering. ■

Combining the ordering \prec_{rpo} for a_i and k_i , and the ordering $\prec_{\mathcal{S}}$ for $|$, \circ , and \bullet , we define the interpretation $[]$ on $(\mathcal{A} \cup \{k_i, i > 0\})^* \times (\mathcal{A} \cup \{\circ, \bullet, |\})^*$ as follows:

$$\begin{aligned} [a_i] &= (u, v) \mapsto (a_i u, a_i v) \\ [k_{i+1}] &= (u, v) \mapsto (\psi_{i+1}(u), v) \\ [\bullet] &= (u, v) \mapsto (u, \bullet v) \\ [\circ] &= (u, v) \mapsto (u, \circ v) \\ [|] &= (u, v) \mapsto (u, |v). \end{aligned}$$

$(\mathcal{A} \cup \{k_i, i > 0\})^* \times (\mathcal{A} \cup \{\circ, \bullet, |\})^*$ is ordered by the left-to-right lexicographic combination of \prec_{rpo} and \prec_S . We finally define \prec by

$$u \prec v \Leftrightarrow [u] \text{lex}(\prec_{rpo}, \prec_S) [v].$$

LEMMA 4.3.

- (i) \prec has the subterm property,
- (ii) \prec is monotone,
- (iii) $\forall i \in \mathbb{N}, \forall l \rightarrow r \in \mathcal{R}_i, \forall w \in \mathcal{A}^*, lw \succ rw$.

Proof. (i) and (ii) are consequences of Lemma 4.1(iv), (v). We establish (iii): We examine each rule of \mathcal{R}_0 and \mathcal{R}_{i+1} and verify that it reduces under the interpretation $[\]$ with the ordering \prec . Let $w \in \mathcal{A}^*$ and $(u, v) = [w]$.

For \mathcal{R}_0 :

$$\begin{aligned} (0, 1): & (a_0u, a_0v) \succ (u, \circ v), \\ (0, 2): & (u, \circ v) \succ (u, \bullet | v), \\ (0, 3): & (u, \bullet | v) \succ (u, | \bullet \bullet v), \\ (0, 4): & (u, | \circ v) \succ (u, \circ | v), \\ (0, 5): & (u, \bullet \bullet v) \succ (u, v). \end{aligned}$$

For \mathcal{R}_{i+1} :

$$\begin{aligned} (i+1, 1): & (a_{i+1}a_0u, \bullet a_{i+1}a_0v) \succ (\psi_{i+1}(a_{i+1}u), a_{i+1}v), \\ (i+1, 2): & (a_{i+1}u, \bullet a_{i+1}v) \succ (a_{i+1}u, a_{i+1} \bullet v), \\ (i+1, 3): & (\psi_{i+1}(u), \bullet v) \succ (\psi_{i+1}(a_iu), a_iv), \\ (i+1, 4): & (a_{i+1}u, a_{i+1} \circ v) \succ (a_{i+1}u, \circ a_{i+1}v), \\ (i+1, 5): & (\psi_{i+1}(u), v) \succ (u, \circ v), \\ (i+1, 6): & (a_{i+1}u, \bullet a_{i+1}v) \succ (\psi_{i+1}(u), v). \end{aligned}$$

(See Lemma 4.1(i), (ii), (iii)). ■

PROPOSITION 4.2. For each $i \in \mathbb{N}$, \mathcal{R}_i is totally terminating.

Proof. Consequence of Lemma 4.3 and Proposition 2.1. ■

5. CONCLUSION

We have proven that it is possible to exhaust the multiple recursive functions using SRS reducing under Higman's lemma. For that, we have encoded the maximal order type of the division ordering Θ by strings equipped with the recursive path ordering. Here is the key point of the construction. Then it is easy to simulate the Hardy hierarchy for this notation system. We would like to mention that the problem was solved using pure logical arguments.

What about *term* rewriting systems? We believe that our approach would apply to term rewriting systems, using the lexicographic path ordering on terms instead of the recursive path ordering on strings: the order type of the lexicographic path ordering reaches the maximal order type of the homeomorphic embedding of Kruskal's theorem.

ACKNOWLEDGMENTS

The author thanks the anonymous referees for fruitful comments.

REFERENCES

1. Cichon, E. A. (1983), A short proof of two recently discovered independence results using recursion theoretic methods, in "Proceedings of the American Mathematical Society," Vol. 97, pp. 704–706, Amer. Math. Soc., Providence, RI.
2. Cichon, E. A., and Tahhan Bittar, E. (1998), Ordinal recursive bounds for Higman's theorem, *Theoret. Comput. Sci.* **201**(12), 63–84.
3. De Jongh, D. H. J., and Parikh, R. (1977), Well-partial orderings and hierarchies, *Indag. Math.* **14**, 195–207.
4. Dershowitz, N. (1982), Orderings for term rewriting systems, *Theoret. Comput. Sci.*, **17**(3), 279–301.
5. Dershowitz, N., and Jouannaud, J. P. (1990), Rewrite systems, in "Handbook of Theoretical Computer Science," Vol. B, North-Holland, Amsterdam.
6. Ferreira, M. C. F., and Zanema, H. (1993), Total termination of term rewriting, in "Proceedings of RTA-93," Lecture Notes in Computer Science, Vol. 690, pp. 213–227, Springer-Verlag, Berlin/New York.
7. Higman, G. (1952), Ordering by divisibility in abstract algebras, *Bull. London Math. Soc.* **3**(2), 326–336.
8. Hofbauer, D., and Lautemann, C. (1989), Termination proofs and the length of derivations, in "Proceedings of RTA-88," Lecture Notes in Computer Science, Vol. 355, pp. 167–177, Springer-Verlag, Berlin/New York.
9. Hofbauer, D. (1992), Termination proofs with multiset path orderings imply primitive recursive derivation lengths, *Theoret. Comput. Sci.* **105**(1), 129–140.
10. Martin, U., and Scott, E. A. (1993), The order types of termination orderings on monadic terms, strings and multisets, in "8th Annual Symposium on Logic in Computer Science," pp. 356–363, IEEE Press, New York.
11. Péter, R. (1967), "Recursive Functions," Academic Press, San Diego.
12. Robbin, J. W. (1965), "Subrecursive Hierarchies," Ph.D., Princeton University.
13. Touzet, H. (1998), A complex example of a simplifying rewrite system, in "Proceedings of ICALP'98," Lecture Notes in Computer Science, Vol. 1442, pp. 507–517, Springer-Verlag, Berlin/New York.
14. Touzet, H. (1998), Encoding the Hydra Battle as a rewrite system, in "Proceedings of MFCS'98," Lecture Notes in Computer Science, Vol. 1450, pp. 267–276, Springer-Verlag, Berlin/New York.