Unbounded recursion and non-size-increasing functions

S. Mazzanti[1]

Dipartimento di Culture del Progetto
Università Iuav di Venezia
Fondamenta delle Terese 2206,
30123 Venezia, Italy

Abstract

We investigate the computing power of function algebras defined by means of unbounded recursion on notation. We introduce two function algebras which contain respectively the regressive logspace computable functions and the non-size-increasing logspace computable functions. However, such algebras are unlikely to be contained in the set of logspace computable functions because this is equivalent to $\mathbf{L} = \mathbf{P}$. Finally, we introduce a function algebra based on simultaneous recursion on notation for the non-size-increasing functions computable in polynomial time and linear space.

Keywords: recursion on notation, logspace computable function, polynomial time computable function.

1 Introduction

Since the introduction of the Grzegorczyk's hierarchy, algebras of subrecursive functions have been defined by means of bounded recursion schemes, and from the early 1990s also by means of predicative recursion schemes.

However, both bounded and predicative recursion do not correspond to programming constructs of real-world programming languages, inhibiting the benefits of a major integration of complexity theory and programming language theory [4,5].

Function algebras mostly characterize function complexity classes containing polynomial growth functions, and either bounded recursion or predicative recursion is needed to avoid functions with exponential growth. On the other hand, computational complexity is mainly concerned with decision problems represented as languages or more generally as relations over a suitable data type.

In [7], the relations on natural numbers decidable in linear space and the languages decidable in logarithmic space have been characterized by means of two function algebras containing very simple base functions and closed with respect to substitution and unbounded simultaneous recursion.

[1] Email: mazzanti@iuav.it

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

One algebra contains a set of regressive functions computable in linear space; the other contains a set of non-size-increasing functions computable in logarithmic space. [2]

However, they contain all the characteristic functions of the decision problems solvable in linear and logarithmic space, respectively.

These results are instances of a general fact: if the base functions of a function algebra do not increase too rapidly, unbounded recursion can be used to describe interesting complexity classes. For example, both regressive and non-size-increasing functions are closed with respect to substitution and recursion on notation.

In [2], Predecessor Machines (Register Machines without increment instructions) have been introduced as a computation model for regressive functions. In particular, it has been shown that the total functions computable by Predecessor Machines are the regressive functions of Grzegorczyk's class $\mathcal{E}_2$.

Type systems for the non-size-increasing polynomial time functions have been extensively studied, see e.g. [3]. Such systems constitute an alternative to predicative recursion which rules out not only recursive definitions leading to exponential growth, but also natural algorithms computing non-size-increasing functions.

Therefore, it seems that regressive and non-size-increasing functions could play a remarkable role in computational complexity.

Let $\mathbf{R}(\mathbf{F})$ be the set of number-theoretic functions defined as the closure with respect to substitution and recursion on notation of a list $\mathbf{F}$ of basic functions, the constant functions and the projection functions.

Algebra $\mathbf{R}(P)$ where P is the predecessor function, has been studied in [8,9] following [7]. Algebra $\mathbf{R}(P)$ is a set of regressive logspace computable functions which contains all the sharply bounded logspace computable functions. [3] Moreover, $\mathbf{R}(P)$ is the set of functions computable by Regressive Machines, which are a polynomial time version of structured Predecessor Machines [2].

In this paper we study new function algebras which extend algebra $\mathbf{R}(P)$ in order to shed some light on some open questions suggested in [9]. For instance, we do not know whether $\mathbf{R}(P)$ equals the set of regressive logspace computable functions.

In Section 3, we consider the algebra $\mathbf{R}(bs_0, bs_1)$ where $s_i(x) = 2x + i$ and $bs_i(x, y) = s_i(x)$ if $s_i(x) \leq y$ and $bs_i(x, y) = x$ otherwise. Algebra $\mathbf{R}(bs_0, bs_1)$ contains the regressive logspace computable functions, but we give strong evidence that $\mathbf{R}(bs_0, bs_1)$ is unlikely to be contained in the set $\mathbf{FL}$ of logspace computable functions because $\mathbf{R}(bs_0, bs_1) \subseteq \mathbf{FL}$ is equivalent to $\mathbf{L} = \mathbf{P}$. In the same way, we show that if $\mathbf{R}(P)$ contained all the regressive logspace computable functions then $\mathbf{L} = \mathbf{P}$ would be true.

In Section 4, we consider the algebra $\mathbf{R}(sbs_0, sbs_1)$ where $sbs_i(x, y) = s_i(x)$ if $|s_i(x)| \leq |y|$ and $sbs_i(x, y) = x$ otherwise. Algebra $\mathbf{R}(sbs_0, sbs_1)$ contains the non-size-increasing logspace computable functions and coincides with algebra $\mathbf{R}(P)$ extended with a concatenation recursion on notation operator.

[2] The values of a regressive function are bounded by the maximum of its arguments and a constant whereas the size of the values of a non-size-increasing function is bounded by the maximum of the sizes of its arguments and a constant.

[3] A function is sharply bounded iff its values can be stored in logarithmic space.

Finally, in Section 5 we consider algebra $\mathbf{S}(sbs_0, sbs_1)$ obtained from $\mathbf{R}(sbs_0, sbs_1)$ by replacing recursion on notation with simultaneous recursion on notation and show that it coincides with the set of non-size-increasing functions computable in polynomial time and linear space. Using the techniques of [9], one can show that $\mathbf{R}(P) = \mathbf{S}(P)$, but at present we do not know whether $\mathbf{R}(sbs_0, sbs_1) = \mathbf{S}(sbs_0, sbs_1)$.

A possible continuation of this work could address such question as well as the search of function algebras for the regressive and non-size-increasing functions of well-known complexity classes.

2 Preliminaries

In this paper, we will only consider functions with finite arity on the set $\mathbb{N} = \{0, 1, \ldots\}$ of natural numbers.

From now on, we agree that $x, y, z, u, v, w, i, j, k, l, n, m$ range over $\mathbb{N}$, that a, b, c range over positive integers, that $\mathbf{x}, \mathbf{y}, \mathbf{z}$ range over sequences (of fixed length) of natural numbers, that p, q, r range over integer polynomials with non negative coefficients (unless otherwise stated) and that f, g, h range over functions.

A function f is a *polynomial growth function* iff there is a polynomial p *majorizing (the length of)* f, i. e. such that $|f(\mathbf{x})| \leq p(|\mathbf{x}|)$ or, equivalently, $f(\mathbf{x}) < 2^{p(|\mathbf{x}|)}$ for any $\mathbf{x}$, where $|x_1, \ldots, x_n|$ is the sequence $|x_1|, \ldots, |x_n|$ and $|x| = \lceil \log_2(x+1) \rceil$ is the number of bits of the binary representation of x. Moreover, f is *sharply bounded* iff there is a polynomial p such that $f(\mathbf{x}) \leq p(|\mathbf{x}|)$ for any $\mathbf{x}$, f is *regressive* iff there is some constant k such that $f(\mathbf{x}) \leq \max(\mathbf{x}, k)$ for any $\mathbf{x}$, see [2] and f is *non-size-increasing* iff there is some constant k such that $|f(\mathbf{x})| \leq \max(|\mathbf{x}|, k)$ for any $\mathbf{x}$. We denote **REG** the set of regressive functions and **NSI** the set of non-size-increasing functions.

We will use the following unary functions: the *binary successor* functions $s_0 : x \mapsto 2x$ and $s_1 : x \mapsto 2x + 1$; for $i \in \{0, 1\}$ the *bounded binary successor* functions $bs_i : x, y \mapsto s_i(x)$ if $s_i(x) \leq y$ and $bs_i : x, y \mapsto x$ otherwise; for $i \in \{0, 1\}$ the *size bounded binary successor* functions $sbs_i : x, y \mapsto s_i(x)$ if $|s_i(x)| \leq |y|$ and $sbs_i : x, y \mapsto x$ otherwise; the *constant* functions $C_y : x \mapsto y$ for any $y \in \mathbb{N}$; the *signum* function $sg : x \mapsto \min(x, 1)$; the *cosignum* function $cosg : x \mapsto 1 - sg(x)$; the *length* function $len : x \mapsto |x|$.

We will also use the following functions: the *modified subtraction* function $sub : x, y \mapsto x \dot{-} y = \max(x - y, 0)$; the *predecessor* function $P : x \mapsto x \dot{-} 1$; the *quadratum* function $quad : x \longmapsto x^2$; the *division* function $quot : x, y \longmapsto \lfloor x/y \rfloor$; the *remainder* function $rem : x, y \longmapsto x - y \lfloor x/y \rfloor$; the *conditional* function $cond : x, y, z \mapsto y$ if $x = 0$ and $cond : x, y, z \mapsto z$ otherwise; the *bit* function $bit : x, y \mapsto rem(\lfloor x/2^y \rfloor, 2)$; the *most significant part* function $MSP : x, y \mapsto \lfloor x/2^y \rfloor$; the *least significant part* function $LSP : x, y \longmapsto x \bmod 2^y$.

Finally, we will use the following operators on functions:

- the *recursion on notation* operator $RN(g, h_0, h_1)$ transforming functions $g : \mathbb{N}^a \to \mathbb{N}$, $h_0 : \mathbb{N}^{a+2} \to \mathbb{N}$ and $h_1 : \mathbb{N}^{a+2} \to \mathbb{N}$ into the function $f : \mathbb{N}^{a+1} \to \mathbb{N}$

such that

$$f(0, \mathbf{y}) = g(\mathbf{y}),$$
$$f(s_i(x), \mathbf{y}) = h_i(x, \mathbf{y}, f(x, \mathbf{y}))$$

where $i \in \{0, 1\}$ and $x > 0$ when $i = 0$. Moreover, the *bounded recursion on notation* $BRN(g, h_0, h_1, l)$ of g, h_0, h_1, l is the function f such that $f = RN(g, h_0, h_1)$ and $f(x, \mathbf{y}) \leq l(x, \mathbf{y})$;

- the *simultaneous recursion on notation* operator $SRN(g_1, \ldots, g_b, h_1^0, h_1^1, \ldots, h_b^0, h_b^1)$ transforming functions $g_1, \ldots, g_b : \mathbb{N}^a \to \mathbb{N}$ and $h_1^0, h_1^1, \ldots, h_b^0, h_b^1 : \mathbb{N}^{a+b+1} \to \mathbb{N}$ into the functions $f_1, \ldots, f_b : \mathbb{N}^{a+1} \to \mathbb{N}$ such that

$$f_j(0, \mathbf{y}) = g_j(\mathbf{y}),$$
$$f_j(s_i(x), \mathbf{y}) = h_i^j(x, \mathbf{y}, f_1(x, \mathbf{y}), \ldots, f_b(x, \mathbf{y}))$$

where $1 \leq j \leq b$, $i \in \{0, 1\}$ and $x > 0$ when $i = 0$;

- the *substitution* operator $SUBST(g_1, \ldots, g_b, h)$ transforming functions $g_1, \ldots, g_b : \mathbb{N}^a \to \mathbb{N}$ and function $h : \mathbb{N}^b \to \mathbb{N}$ into the function $f : \mathbb{N}^a \to \mathbb{N}$ such that $f(\mathbf{x}) = h(g_1(\mathbf{x}), \ldots, g_b(\mathbf{x}))$.

For any function $f : \mathbb{N}^a \to \mathbb{N}$, set $bit_f(\mathbf{x}, i) = bit(f(\mathbf{x}), i)$ and

$$f(\mathbf{x})[i, j] = MSP(LSP(f(x, \mathbf{y}), i + 1), j) = \sum_{j \leq k \leq i} bit_f(\mathbf{x}, k) 2^{k-j}.$$

Note that $f(\mathbf{x})[i, j]$ is the number with binary representation the bits from position j to position i of $f(\mathbf{x})$.

As usual, the characteristic function of a predicate Q on natural numbers is the function $f(\mathbf{x})$ returning 1 if $Q(\mathbf{x})$ is true, 0 otherwise.

For any functions $f_1, \ldots, f_a$, any sets $\mathbf{F_1}, \ldots, \mathbf{F_b}$ of functions and any set $\{op_1, \ldots, op_c\}$ of operators, let $\text{clos}(f_1, \ldots, f_a, \mathbf{F_1}, \ldots, \mathbf{F_b}; op_1, \ldots, op_c)$ be the inductive closure of $\{f_1, \ldots, f_a\} \cup \mathbf{F_1} \cup \ldots \cup \mathbf{F_b} \cup \mathbf{I}$ with respect to $\{op_1, \ldots, op_c\}$ where $\mathbf{I}$ is the set of the projection functions $I^a[i] : x_1, \ldots, x_a \longmapsto x_i$ with any arity a and $1 \leq i \leq a$.

Finally, let

$$\mathbf{R}(\mathbf{F}) = \text{clos}(\mathbf{F}, \{C_n\}_n; SUBST, RN)$$

and

$$\mathbf{S}(\mathbf{F}) = \text{clos}(\mathbf{F}, \{C_n\}_n; SUBST, SRN).$$

By definition, we have that $\mathbf{R}(\mathbf{F}) \subseteq \mathbf{S}(\mathbf{F})$. We set $\mathbf{R}(f_1, \ldots, f_a) = \mathbf{R}(\{f_1, \ldots, f_a\})$.

Lemma 2.1 *If* $\mathbf{F}$ *is a set of regressive (non-size-increasing) functions, then also* $\mathbf{R}(\mathbf{F})$ *and* $\mathbf{S}(\mathbf{F})$ *are sets of regressive (non-size-increasing) functions.*

Lemma 2.2 *If* $\mathbf{F}$ *is a set of polynomial time regressive (non-size-increasing) functions, then also* $\mathbf{R}(\mathbf{F})$ *and* $\mathbf{S}(\mathbf{F})$ *are sets of polynomial time regressive (non-size-increasing) functions.*

The following results concerning class $\mathbf{R}(P)$ have been obtained in [8,9]:

- $\mathbf{R}(P) \subseteq \mathbf{FL}$;
- the sharply bounded logspace computable functions coincide with the sharply bounded functions in $\mathbf{R}(P)$;
- the characteristic functions of logspace predicates coincide with the $0-1$ valued functions in $\mathbf{R}(P)$.

We do not know whether $\mathbf{R}(P)$ equals the set of regressive logspace computable functions.

3 Recursion on notation and regressive functions

In this section we study algebra $\mathbf{R}(bs_0, bs_1)$. First, we show that $\mathbf{R}(bs_0, bs_1)$ contains all the logspace computable regressive functions.

Then, we give strong evidence that $\mathbf{R}(bs_0, bs_1) \subseteq \mathbf{FL}$ is not true. Indeed, $\mathbf{R}(bs_0, bs_1)$ contains a function which can decide the iterated mod problem [6], a $\mathbf{P}$-complete problem under logspace reductions. Thus we may conclude that $\mathbf{R}(bs_0, bs_1) \subseteq \mathbf{FL}$ is equivalent to $\mathbf{L} = \mathbf{P}$.

In the same way, we show that if $\mathbf{R}(P)$ contained all the regressive logspace computable functions then $\mathbf{L} = \mathbf{P}$ would be true.

We start by noting that, from the results of [8,9] listed at the end of the Preliminaries, class $\mathbf{R}(bs_0, bs_1)$ contains all the regressive logspace computable functions.

Theorem 3.1 $\mathbf{FL} \cap \mathbf{REG} \subseteq \mathbf{R}(bs_0, bs_1)$.

Proof. First, $P \in \mathbf{R}(bs_0, bs_1)$ because

$$P(0) = 0, P(s_0(x)) = s_1(P(x)), P(s_1(x)) = s_0(x)$$

and $P(x) = p(x, x)$ where

$$\begin{aligned} p(0, y) &= 0, \\ p(s_0(x), y) &= bs_1(p(x, y), y), \\ p(s_1(x), y) &= bs_0(x, y). \end{aligned}$$

Assume that $f \in \mathbf{FL} \cap \mathbf{REG}$. Then $f(\mathbf{x}) \leq \max(\mathbf{x}, k)$ for some k and $bit_f(\mathbf{x}, |\max(\mathbf{x}, k)| \dot{-} |s_i(u)|) \in \mathbf{R}(P) \subseteq \mathbf{R}(bs_0, bs_1)$ because it is a $0-1$ valued logspace computable function. Thus we compute $f(\mathbf{x})$ by concatenating its bits. To this aim we define the function

$$\begin{aligned} g(0, \mathbf{x}) \quad &= 0\,, \\ g(s_i(u), \mathbf{x}) \; &= bs_{bit_f(\mathbf{x}, |\max(\mathbf{x},k)| \dot{-} |s_i(u)|)}(g(u, \mathbf{x}), \max(\mathbf{x}, k)) \end{aligned}$$

such that $g(\max(\mathbf{x}, k), \mathbf{x}) = f(\mathbf{x})$ and so $f \in \mathbf{R}(bs_0, bs_1)$. □

One could ask whether $\mathbf{R}(bs_0, bs_1) \subseteq \mathbf{FL}$ since it is obtained by adding the bounded successors to $\mathbf{R}(P)$. Below, we show that this is very unlikely to be

true. Indeed, we will show that the iterated mod problem [6], which is a **P**-complete problem under logspace reductions, is decidable in $\mathbf{R}(bs_0, bs_1)$ and so $\mathbf{R}(bs_0, bs_1) \subseteq \mathbf{FL}$ is equivalent to $\mathbf{L} = \mathbf{P}$.

The iterated mod problem asks if the sequence of remainders $rem(\ldots rem(rem(a, b_1), b_2), \ldots, b_n)$ is zero or not, where $a, b_1, \ldots, b_n$ is a sequence of positive integers. We consider the coding C of number sequences of any length such that $C(b_1, \ldots, b_n)$ is the number whose binary representation is $10\overline{b_n} \ldots 10\overline{b_1}$ where $\overline{m}$ is the bit string obtained by doubling the bits of m: $\overline{0} = \lambda$, $\overline{s_i(m)} = \overline{m}ii$. The following theorem shows that there is a function $itm \in \mathbf{R}(bs_0, bs_1)$ such that

$$itm(a, C(b_1, \ldots, b_n)) = cosg(rem(\ldots rem(rem(a, b_1), b_2), \ldots, b_n)),$$

i.e. itm decides the iterated mod problem.

Theorem 3.2 *There is a function $itm \in \mathbf{R}(bs_0, bs_1)$ which decides the iterated mod problem.*

Proof. By Theorem 3.1, $cosg, rem \in \mathbf{R}(bs_0, bs_1)$. Now we show that there are functions $tail, head, l$ in $\mathbf{R}(bs_0, bs_1)$ such that $tail(C(b_1, \ldots, b_{n+1})) = C(b_1, \ldots, b_n)$, $head(C(b_1, \ldots, b_n)) = b_n$, and $l(C(b_1, \ldots, b_n)) = n$.

First, let

$$\begin{aligned} t(0, b) &= b\,, \\ t(s_i(x), b) &= \begin{cases} div(t(x, b), 4) & \text{if } bit(t(x, b), 0) = bit(t(x, b), 1) \\ t(x, b) & \text{otherwise} \end{cases} \end{aligned}$$

and note that $t \in \mathbf{R}(P) \subseteq \mathbf{FL}$ and the string $10\overline{b_1} \ldots 10\overline{b_n}10$ is the binary representation of $t(C(b_1, \ldots, b_{n+1}), C(b_1, \ldots, b_{n+1}))$.

Then $tail(b) = div(t(b, b), 4)$ and $head(b) = extr_2(LSP(b, |b| \dot{-} |t(b, b)|))$ belong to $\mathbf{R}(bs_0, bs_1)$ where $extr_2 : \sum_{i \le n} x_i 2^i \mapsto \sum_{i \le \lfloor n/2 \rfloor} x_{2i} 2^i$ is in $\mathbf{R}(bs_0, bs_1)$ by Theorem 3.1 because $extr_2$ is a regressive logspace computable function. Moreover, l and $x \mapsto 2^{l(x)} - 1$ are in $\mathbf{R}(bs_0, bs_1)$ by Theorem 3.1. Indeed, l is a logspace computable function which counts the occurrences of string "10" in the binary representation of its argument and can be defined by sharply bounded recursion as follows

$$\begin{aligned} l(0) &= 0\,, \\ l(s_i(x)) &= \begin{cases} l(x) + 1 & \text{if } (odd(|x|) \wedge (i = 0) \wedge bit(x, 0) = 1), \\ l(x) & \text{otherwise} \end{cases} \end{aligned}$$

where $l(x) < |x|$ and odd is the characteristic function of odd numbers. Furthermore $2^{l(x)} - 1$ is a regressive function because $2^{l(x)} - 1 \le 2^{|x|-1} - 1 \le x$ and it belongs to $\mathbf{R}(bs_0, bs_1)$ by Theorem 3.1.

Recall that the function $it_g(x, z) = g^{|x|}(z)$ belongs to $\mathbf{R}(bs_0, bs_1)$ for any regressive function $g \in \mathbf{R}(bs_0, bs_1)$.

Then the function $C^-(x,y) = head(it_{tail}(div(x,2),y))$ belongs to $\mathbf{R}(bs_0,bs_1)$ and we have that

$$C^-(x,C(b_1,\ldots,b_n)) = head(tail^{|x|\dot{-}1}(C(b_1,\ldots,b_n))) = b_{|x|}$$

for $1 \leq |x| \leq n$. Therefore, for

$$\begin{aligned} f(0,a,b) &= a\,, \\ f(s_i(x),a,b) &= rem(f(x,a,b),C^-(x,b)), \end{aligned}$$

we obtain

$$f(2^{l(C(b_1,\ldots,b_n))}-1,a,C(b_1,\ldots,b_n)) = rem(\ldots rem(rem(a,b_1),b_2),\ldots,b_n)$$

and $itm(a,b) = cosg(f(2^{l(b)}-1,a,b))$. □

From the theorem above, we obtain immediately the following statement.

Theorem 3.3 $\mathbf{L} = \mathbf{P} \Leftrightarrow \mathbf{R}(bs_0,bs_1) \subseteq \mathbf{FL}$.

Proof. By [6], the iterated mod problem is **P**-complete under logspace reductions. Hence, if $\mathbf{R}(bs_0,bs_1) \subseteq \mathbf{FL}$ then $itm \in \mathbf{L}$ by Theorem 3.2 and so $\mathbf{L} = \mathbf{P}$. On the other hand, if $\mathbf{L} = \mathbf{P}$ then $\mathbf{FL} = \mathbf{FP}$ but by Lemma 2 $\mathbf{R}(bs_0,bs_1) \subseteq \mathbf{FP}$ and therefore $\mathbf{R}(bs_0,bs_1) \subseteq \mathbf{FP} = \mathbf{FL}$. □

Since $\mathbf{R}(P)$ is a set of regressive functions, it is natural to ask if it contains all the regressive logspace computable functions. The same strategy used to show Theorem 3.2 proves that this is unlikely to be true.

Theorem 3.4 *If* $\mathbf{R}(P) = \mathbf{FL} \cap \mathbf{REG}$ *then* $\mathbf{L} = \mathbf{P}$.

Proof. We give only a sketch of the proof. If $\mathbf{R}(P) = \mathbf{FL} \cap \mathbf{REG}$ then the proof of Theorem 3.2 can be adapted to show that $itm \in \mathbf{R}(P)$. But $itm \in \mathbf{R}(P)$ implies $itm \in \mathbf{L}$ and so $\mathbf{L} = \mathbf{P}$ because the iterated mod problem is **P**-complete. □

Corollary 3.5 *If* $\mathbf{R}(P) = \mathbf{FL} \cap \mathbf{REG}$ *then* $\mathbf{R}(P) = \mathbf{FP} \cap \mathbf{REG}$.

Proof. If $\mathbf{R}(P) = \mathbf{FL} \cap \mathbf{REG}$ then $\mathbf{L} = \mathbf{P}$. Thus $\mathbf{FL} = \mathbf{FP}$ and therefore $\mathbf{R}(P) = \mathbf{FP} \cap \mathbf{REG}$. □

4 Recursion on notation and non-size-increasing functions

In this section we show that algebra $\mathbf{R}(sbs_0,sbs_1)$ contains the non-size-increasing logspace computable functions and is the class of non-size-increasing functions obtained by extending algebra $\mathbf{R}(P)$ with a concatenation recursion on notation operator.

Finally, we show that $\mathbf{R}(sbs_0,sbs_1) \subseteq \mathbf{FL}$ is equivalent to $\mathbf{L} = \mathbf{P}$.

Consider the *simple concatenation recursion on notation* operator $SCRN(h_0, h_1)$ transforming $0-1$ valued functions $h_0, h_1 : \mathbb{N}^{a+1} \to \mathbb{N}$ into the function $f : \mathbb{N}^{a+1} \to \mathbb{N}$ such that

$$\begin{aligned} f(0, \mathbf{y}) &= 0\,, \\ f(s_i(x), \mathbf{y}) &= s_{h_i(x,\mathbf{y})}(f(x, \mathbf{y})) \end{aligned}$$

($i \in \{0, 1\}$ and $x > 0$ when $i = 0$) and set

$$\mathbf{CR}(\mathbf{F}) = \mathrm{clos}(\mathbf{F}, \{C_n\}_n; SUBST, SCRN, RN).$$

We start by noting that class $\mathbf{CR}(P)$ contains all the non-size-increasing logspace computable functions.

Theorem 4.1 $\mathbf{FL} \cap \mathbf{NSI} \subseteq \mathbf{CR}(P)$.

Proof. Assume that $f \in \mathbf{FL} \cap \mathbf{NSI}$. Then, $|f(\mathbf{x})| \leq \max(|\mathbf{x}|, k)$ for some k and $bit_f(\mathbf{x}, \max(|\mathbf{x}|, k) \dot{-} |s_i(u)|) \in \mathbf{R}(P) \subseteq \mathbf{CR}(P)$ because it is a $0-1$ valued logspace computable function. Therefore, by using $SCRN$, we define

$$\begin{aligned} g(0, \mathbf{x}) &= 0\,, \\ g(s_i(u), \mathbf{x}) &= s_{bit_f(\mathbf{x}, \max(|\mathbf{x}|,k) \dot{-} |s_i(u)|)}(g(u, \mathbf{x})) \end{aligned}$$

such that $g(u, \mathbf{x}) = f(\mathbf{x})[m, m \dot{-} |u|]$ where $m = \max(|\mathbf{x}|, k)$.

Then $f(\mathbf{x}) = g(\max(\mathbf{x}, 2^k - 1), \mathbf{x})$ and so $f \in \mathbf{CR}(P)$. □

Since the size bounded binary successor functions are non-size-increasing logspace computable functions, we obtain immediately the following corollary.

Corollary 4.2 $sbs_0, sbs_1 \in \mathbf{CR}(P)$.

Now we show that the closure with respect to $SCRN$ is equivalent to adding sbs_0 and sbs_1 to the base functions.

Theorem 4.3 $\mathbf{R}(sbs_0, sbs_1) = \mathbf{CR}(P)$.

Proof. By Corollary 4.2, $\mathbf{R}(sbs_0, sbs_1) \subseteq \mathbf{CR}(P)$. On the other hand, to show that $\mathbf{CR}(P) \subseteq \mathbf{R}(sbs_0, sbs_1)$ it suffices to show that $P \in \mathbf{R}(sbs_0, sbs_1)$ and that $\mathbf{R}(sbs_0, sbs_1)$ is closed with respect to $SCRN$. The proof of $P \in \mathbf{R}(sbs_0, sbs_1)$ is similar to that of $P \in \mathbf{R}(bs_0, bs_1)$ in the proof of Theorem 3.1. As concerns the closure of $\mathbf{R}(sbs_0, sbs_1)$ under $SCRN$, consider the function

$$\begin{aligned} g(0, \mathbf{y}, z) &= 0\,, \\ g(s_i(x), \mathbf{y}, z) &= sbs_{h_i(x,\mathbf{y})}(g(x, \mathbf{y}, z), z) \end{aligned}$$

where $h_0, h_1 : \mathbb{N}^{a+1} \to \mathbb{N}$ are $0-1$ valued functions in $\mathbf{R}(sbs_0, sbs_1)$, with $i \in \{0, 1\}$

and $x > 0$ when $i = 0$. Also, note that

$$sbs_{h_i(x,\mathbf{y})}(u,z) = \begin{cases} sbs_0(u,z) & \text{if } h_i(x,\mathbf{y}) = 0 \\ sbs_1(u,z) & \text{otherwise} \end{cases}$$

belongs to $\mathbf{R}(sbs_0, sbs_1)$. Then, for

$$\begin{aligned} f(0,\mathbf{y}) &= 0\,, \\ f(s_i(x),\mathbf{y}) &= s_{h_i(x,\mathbf{y})}(f(x,\mathbf{y})) \end{aligned}$$

we have $f \in \mathbf{R}(sbs_0, sbs_1)$ because $f(x,\mathbf{y}) = g(x,\mathbf{y},\max(x,\mathbf{y}))$, $\max \in \mathbf{R}(P)$ and $g \in \mathbf{R}(sbs_0, sbs_1)$. Note that it can be shown by induction on x that $f(x,\mathbf{y}) = g(x,\mathbf{y},z)$ for any z such that $\max(|x|,|\mathbf{y}|) \leq |z|$. □

Since $\mathbf{R}(bs_0, bs_1) \subseteq \mathbf{R}(sbs_0, sbs_1)$, it is easy to see that Theorem 3.3 holds also for $\mathbf{R}(sbs_0, sbs_1)$.

Theorem 4.4 $\mathbf{L} = \mathbf{P} \Leftrightarrow \mathbf{R}(sbs_0, sbs_1) \subseteq \mathbf{FL}$.

5 Simultaneous recursion and non-size-increasing functions

In this section we show that the class $\mathbf{S}(sbs_0, sbs_1)$ coincide with the class of non-size-increasing functions computable in polynomial time and linear space.

We start by claiming that a statement analogous to Theorem 4.3 holds for algebra $\mathbf{S}(sbs_0, sbs_1)$. A proof can be derived from that of Theorem 4.3.

Theorem 5.1 $\mathbf{S}(sbs_0, sbs_1) = \mathrm{clos}(P, \{C_n\}_n; SUBST, SCRN, SRN)$.

The following theorem recalls a function algebra for the functions computable in polynomial time and linear space, see [1, Theorem 3.45].

Theorem 5.2

$$\mathbf{FPTIMELINSPACE} = \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN).$$

Now we show that $\mathbf{S}(sbs_0, sbs_1)$ is a subset of the function algebra of Theorem 5.2.

For $c, m > 0$, let $\langle x_c, \ldots, x_1; m\rangle = \sum_{i<c} x_{i+1} 2^{mi}$. If $x_c, \ldots, x_1 < 2^m$ then $x_c, \ldots, x_1$ are the base 2^m digits of $\langle x_c, \ldots, x_1; m\rangle$.

Theorem 5.3 $\mathbf{S}(sbs_0, sbs_1) \subseteq \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN)$.

Proof. Set $\mathbf{C} = \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN)$. We show the theorem by induction on $\mathbf{S}(sbs_0, sbs_1)$.

The induction basis and the induction step concerning substitution are trivial. In the following we sketch the proof of the induction step concerning simultaneous recursion on notation.

Let $f_1, \ldots, f_c$ be the functions defined by simultaneous recursion on notation from $g_1, \ldots, g_c, h_1^0, h_1^1, \ldots, h_c^0, h_c^1 \in \mathbf{S}(sbs_0, sbs_1)$.

Let $m = \max(|x, \mathbf{y}|, b)$ and assume that $|f_i(x, \mathbf{y})| \leq m$. We define a function $f \in \mathbf{C}$ such that $f(x, \mathbf{y})$ encodes $f_1(x, \mathbf{y}), \ldots, f_c(x, \mathbf{y})$ with a single number of cm bits, i.e. $f(x, \mathbf{y}) = \langle f_1(x, \mathbf{y}), \ldots, f_c(x, \mathbf{y}); m\rangle$:

$$\begin{aligned} f(0, \mathbf{y}) &= \langle g_c(x, \mathbf{y}), \ldots, g_1(x, \mathbf{y}); m\rangle\,, \\ f(s_i(x), \mathbf{y}) &= \left\langle h_1^i(z_1, \ldots, z_c), \ldots, h_c^i(z_1, \ldots, z_c); m\right\rangle \end{aligned}$$

where $z_j = f(x, \mathbf{y})[jm - 1, (j-1)m]$ for $1 \leq j \leq c$ and $|f(x, \mathbf{y})| \leq c \cdot \max(|x, \mathbf{y}|, b)$. Then $f_j \in \mathbf{C}$ because $f_j(x, \mathbf{y}) = f(x, \mathbf{y})[jm - 1, (j-1)m]$. □

The next lemma states that the class $\mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN)$ contains linear growth functions only.

Lemma 5.4 *For any $f \in \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN)$ there are b_0 and c_0 such that for any $b \geq b_0$ and $c \geq c_0$ we have $|f(\mathbf{x})| \leq c \cdot \max(|\mathbf{x}|, b)$.*

If $|f(\mathbf{x})| \leq c \cdot \max(|\mathbf{x}|, b)$ then we say that b and c bound f. The following lemma is the bulk of the proof that

$$\mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN) \cap \mathbf{NSI} \subseteq \mathbf{S}(sbs_0, sbs_1).$$

It states that any function $f \in \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN)$ can be expressed as the concatenation of c functions in $\mathbf{S}(sbs_0, sbs_1)$ where c is a constant depending on f. The basic idea of the proof is the same as [7, Lemma 3.4]: representing a block of cn bits as c blocks of n bits.

Lemma 5.5 *For any $f \in \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN)$ with arity a there are b_0 and c_0 such that $|f(\mathbf{x})| \leq c_0 \cdot \max(|\mathbf{x}|, b_0)$ and for any $c \geq c_0$ there are functions $f_1, \ldots, f_c : \mathbb{N}^{ca+1} \to \mathbb{N}$ belonging to $\mathbf{S}(sbs_0, sbs_1)$ such that if $c_0 \cdot \max(|\mathbf{x}|, b_0) \leq c|n|$, then for any $i \in \{1, \ldots, c\}$,*

$$f_i(x_{1c}, \ldots, x_{11}, \ldots, x_{ac}, \ldots, x_{a1}, n) = f(\mathbf{x})[i|n| - 1, (i-1)|n|]$$

where $x_j = \langle x_{jc}, \ldots, x_{j1}; |n|\rangle$ and $x_{jk} < 2^{|n|}$ for $1 \leq j \leq a$ and $1 \leq k \leq c$.

Proof. We show the lemma by induction on

$$\mathbf{C} = \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN).$$

Note that Lemma 5.4 states that for any $f \in \mathbf{C}$ there are infinite pairs (b_0, c_0) which bound f. For $1 \leq i \leq a$, let $\mathbf{x}_i = x_{ic}, \ldots, x_{i1}$ be the sequence of digits in base $2^{|n|}$ of x_i.

Induction basis. Let f be any initial function of $\mathbf{C}$ and consider any constants b_0 and c_0 which bound f. Since $f \in \mathbf{FL}$, also

$$f'(\mathbf{x}_1, \ldots, \mathbf{x}_a, n) = f(\langle x_{1c}, \ldots, x_{11}; |n|\rangle, \ldots \langle x_{ac}, \ldots, x_{a1}; |n|\rangle) = f(\mathbf{x})$$

belongs to $\mathbf{FL}$ and $bit_{f'} \in \mathbf{S}(sbs_0, sbs_1)$ by Theorems 4.1 and 4.3.

Assume $c_0 \cdot \max(|\mathbf{x}|, b_0) \leq c|n|$ so that $|f(\mathbf{x})| \leq c_0 \cdot \max(|\mathbf{x}|, b_0) \leq c|n|$. Then for $f_i(\mathbf{x}_1, \ldots, \mathbf{x}_a, n) = f'(\mathbf{x}_1, \ldots, \mathbf{x}_a, n)[i|n| - 1, (i-1)|n|]$ where $1 \leq i \leq c$, we have $f_i(\mathbf{x}_1, \ldots, \mathbf{x}_a, n) = f(\mathbf{x})[i|n| - 1, (i-1)|n|]$.

By Theorem 5.1 $f_i \in \mathbf{S}(sbs_0, sbs_1)$ because $f_i \in \mathbf{FL} \cap \mathbf{NSI}$.

Induction step. Let $f(\mathbf{x}) = h(g(\mathbf{x}))$ and assume that there are b_1 and c_1 bounding g and b_2 and c_2 bounding h (to keep notation simple we consider the case of a single function g). Then for $b_0 = \max(b_1, b_2)$ and $c \geq c_0 = c_1 \cdot c_2$ it is easy to see that $f_i(\mathbf{x}_1, \ldots, \mathbf{x}_a, n) = h_i(g_c(\mathbf{x}_1, \ldots, \mathbf{x}_a, n), \ldots, g_1(\mathbf{x}_1, \ldots, \mathbf{x}_a, n), n)$ satisfies the lemma.

Now, consider function f defined by bounded recursion on notation

$$\begin{aligned} f(0, y) &= g(y)\,, \\ f(s_i(x), y) &= h_i(x, y, f(x, y)) \end{aligned}$$

where $f(x, y) \leq p(x, y)$ for some polynomial p, or in other words, $|f(x, y)| \leq c_0 \cdot \max(|x|, |y|, b_0)$ (we consider the case of a single parameter y to keep notation simple). We also assume that b_0 and c_0 bound g, h_0 and h_1.

Then, for $c \geq c_0$ there are functions $g_1, \ldots, g_c$ and functions $h_1^0, h_1^1, \ldots, h_c^0, h_c^1$ such that

$$g_i(\mathbf{y}, m) = g(y)[i|m| - 1, (i-1)|m|]$$

where $c_0 \cdot \max(|y|, b_0) \leq c|m|$ and $\mathbf{y} = y_c, \ldots, y_1$ is the sequence of digits in base $2^{|m|}$ of y, and

$$h_i^j(\mathbf{x}, \mathbf{y}, \mathbf{z}, r) = h_j(x, y, z)[i|r| - 1, (i-1)|r|]$$

where $c_0 \cdot \max(|x|, |y|, |z|, b_0) \leq c|r|$ and $\mathbf{x} = x_c, \ldots, x_1$, $\mathbf{y} = y_c, \ldots, y_1$ and $\mathbf{z} = z_c, \ldots, z_1$ are the sequences of digits in base $2^{|r|}$ of x, y, z, respectively.

Furthermore, for $1 \leq i, k \leq c$ and $j \in \{0, 1\}$, we define functions $h_{i,k}^j$ such that

$$h_{i,k}^j(u, \mathbf{x}, \mathbf{y}, \mathbf{z}, r) = h_i^j(0, \ldots, 0, MSP(x_c, |r| \dot{-} |u|), x'_1, \ldots, x'_{k-1}, \mathbf{y}, \mathbf{z}, r)$$

where

$$x'_l = LSP(x_{c-(l-1)}, |r| \dot{-} |u|) * MSP(x_{c-l}, |r| \dot{-} |u|)$$

for $1 \leq l < k$ and $v * w = v \cdot 2^{|w|} + v$.

Note that $LSP(x_{c-(l-1)}, |r| \dot{-} |u|) * MSP(x_{c-l}, |r| \dot{-} |u|) \in \mathbf{S}(sbs_0, sbs_1)$ because it is computable in logarithmic space and its length is bounded by $|r|$. Now, for $1 \leq i, k \leq c$ consider functions $f_{i,k}$ such that

$$\begin{aligned} f_{i,1}(0, \mathbf{x}, \mathbf{y}, n) &= g_i(\mathbf{y}, n)\,, \\ f_{i,1}(s_j(u), \mathbf{x}, \mathbf{y}, n) &= h_{i,1}^j(u, \mathbf{x}, \mathbf{y}, f_{c,1}(u, \mathbf{x}, \mathbf{y}, n), \ldots, f_{1,1}(u, \mathbf{x}, \mathbf{y}, n), n), \end{aligned}$$

and for $1 < k \leq c$,

$$\begin{aligned} f_{i,k}(0, \mathbf{x}, \mathbf{y}, n) &= f_{i,k-1}(n, \mathbf{x}, \mathbf{y}, n)\,, \\ f_{i,k}(s_j(u), \mathbf{x}, \mathbf{y}, n) &= h_{i,k}^j(u, \mathbf{x}, \mathbf{y}, f_{c,k}(u, \mathbf{x}, \mathbf{y}, n), \ldots, f_{1,k}(u, \mathbf{x}, \mathbf{y}, n), n). \end{aligned}$$

Thus, for $1 \leq k \leq c$ and $c_0 \cdot \max(|x|, |y|, b_0) \leq c|n|$ we have $|f_{i,k}(n, \mathbf{x}, \mathbf{y}, n)| \leq |n|$ and

$$f_{i,k}(n, \mathbf{x}, \mathbf{y}, n) = f(\langle x_c, \ldots, x_{c-k+1}; |n| \rangle, y)[i|n| - 1, (i-1)|n|].$$

Finally, $f_i(\mathbf{x},\mathbf{y},n) = f_{i,c}(n,\mathbf{x},\mathbf{y},n) = f(x,y)[i|n|-1,(i-1)|n|]$. □

Theorem 5.6

$$\mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN) \cap \mathbf{NSI} \subseteq \mathbf{S}(sbs_0, sbs_1).$$

Proof. By the lemma above, for any $f \in \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN)$ of arity a there are b_0 and c_0 such that $|f(\mathbf{x})| \leq c_0 \cdot \max(|\mathbf{x}|, b_0)$ and for any $c \geq c_0$ there are functions $f_1, \ldots, f_c : \mathbb{N}^{ca+1} \to \mathbb{N}$ belonging to $\mathbf{S}(sbs_0, sbs_1)$ such that if $c_0 \cdot \max(|\mathbf{x}|, b_0) \leq c|n|$ then for any $i \in \{1, \ldots, c\}$,

$$f_i(x_{1c}, \ldots, x_{11}, \ldots, x_{ac}, \ldots, x_{a1}, n) = f(\mathbf{x})[i|n|-1, (i-1)|n|]$$

where $x_j = \langle x_{jc}, \ldots, x_{j1}; |n|\rangle$ and $x_{jk} < 2^{|n|}$ for $1 \leq j \leq a$ and $1 \leq k \leq c$. But, if f is non-size-increasing there is some $b \geq b_0$ such that $|f(\mathbf{x})| \leq \max(|\mathbf{x}|, b)$ and we obtain $f_1(0, \ldots, 0, x_1, \ldots, 0, \ldots, 0, x_a, \max(\mathbf{x}, 2^b - 1)) = f(\mathbf{x})[m-1, 0] = f(\mathbf{x})$ for $m = \max(|\mathbf{x}|, b)$. □

From Theorem 5.3, Theorem 5.6 and Theorem 5.2 we obtain immediately the following characterization of non-size-increasing polynomial time and linear space computable functions.

Theorem 5.7

$$\begin{aligned}\mathbf{S}(sbs_0, sbs_1) &= \mathrm{clos}(C_0, s_0, s_1, \max, quad; SUBST, BRN) \cap \mathbf{NSI} \\ &= \mathbf{FPTIMELINSPACE} \cap \mathbf{NSI}.\end{aligned}$$

Finally, we have a new characterization of the predicates computable in polynomial time and linear space.

Corollary 5.8 *The characteristic functions of polynomial time and linear space computable predicates coincide with the* $0-1$ *valued functions in* $\mathbf{S}(sbs_0, sbs_1)$.

6 Conclusions

In this paper we have continued the study started in [7,8,9] on the computing power of pure recursion schemes which avoid both syntactical constraints and bounds on (the growth rate of) functions.

We have introduced three new function algebras extending the algebra $\mathbf{R}(P)$ of [8,9] in order to shed some light on some open questions suggested in [9].

First, we have considered the algebra $\mathbf{R}(bs_0, bs_1)$. We have shown that $\mathbf{R}(bs_0, bs_1)$ contains the regressive logspace computable functions, but we have given strong evidence that $\mathbf{R}(bs_0, bs_1)$ is not contained in the set of logspace computable functions because $\mathbf{R}(bs_0, bs_1) \subseteq \mathbf{FL}$ is equivalent to $\mathbf{L} = \mathbf{P}$. We have also shown that if $\mathbf{R}(P)$ contained all the regressive logspace computable functions then $\mathbf{L} = \mathbf{P}$ would be true.

Then, we have considered the algebra $\mathbf{R}(sbs_0, sbs_1)$. We have shown that $\mathbf{R}(sbs_0, sbs_1)$ contains the non-size-increasing logspace computable functions and coincides with algebra $\mathbf{R}(P)$ extended with a concatenation recursion on notation

operator. However, even in this case it is unlikely that $\mathbf{R}(sbs_0, sbs_1) \subseteq \mathbf{FL}$ because $\mathbf{R}(sbs_0, sbs_1) \subseteq \mathbf{FL}$ is equivalent to $\mathbf{L} = \mathbf{P}$.

Finally, we have considered algebra $\mathbf{S}(sbs_0, sbs_1)$ and we have shown that it coincides with the set of non-size-increasing functions computable in polynomial time and linear space. Summarizing the results of this paper, we have

$$\mathbf{R}(bs_0, bs_1) \subsetneq \mathbf{R}(sbs_0, sbs_1) \subseteq \mathbf{S}(sbs_0, sbs_1) = \mathbf{FPTIMELINSPACE} \cap \mathbf{NSI}$$

where the leftmost inclusion is strict because sbs_0 and sbs_1 are not regressive. Using the encoding technique of regressive machines introduced in [9, Section 6], one can show that $\mathbf{R}(P) = \mathbf{S}(P)$, but it does not seem possible that such technique can be applied to show that $\mathbf{R}(sbs_0, sbs_1) = \mathbf{S}(sbs_0, sbs_1)$. A possible continuation of this work could address such question as well as the search of function algebras for the regressive and non-size-increasing functions of well-known complexity classes.

References

[1] P. Clote, *Computation models and function algebras*, in *Handbook of Computability Theory*, ed. E.R. Griffor, Elsevier (1999) 589-681.

[2] P. C. Fischer, J. C. Warkentin, *Predecessor Machines*, J. Comput. System Sci. **8** (1974) 190-219.

[3] M. Hofmann, *Linear types and non-size-increasing polynomial time computation*, Inform. and Comp. **183** (2003) 57-85.

[4] N. D. Jones, *Computability and Complexity from a Programming Perspective*, MIT Press, Cambridge, MA, 1997.

[5] N. D. Jones, *LOGSPACE and PTIME characterized as programming languages*, Theoret. Comput. Sci. 228 (1999) 151-174.

[6] H. J. Karloff, W. L. Ruzzo, *The Iterated Mod Problem*, Inform. and Comp. **80** (1989) 193-204.

[7] L. Kristiansen, *Neat algebraic characterizations of LOGSPACE and LINSPACE*, Comput. Complexity **14** (2005) 72-88.

[8] S. Mazzanti, *Logspace computability and regressive machines*, ICTCS 2014, Ceur Workshop Proceedings **1231** (2014) 285-289, URL: http://ceur-ws.org/Vol-1231/short8.pdf.

[9] S. Mazzanti, *Regressive computations characterize logarithmic space*, URL: http://rice.iuav.it/414/, submitted for publication.