Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Cota superior para la longitud de secuencias malas en el orden mayorante con aplicación a complejidad de problemas de decisión en autómatas sobre árboles

Upper bound for the length of bad sequences in the majoring ordering with an application to complexity of decision problems in tree automata

Tesis presentada para optar al título de
Licenciado en Ciencias de la Computación

Gabriel Ignacio Senno

Director: Santiago Daniel Figueira

Buenos Aires, 14 de agosto de 2012

CONTENTS

ABSTRACT

Well quasi-orders (wqo's) are an important mathematical tool for proving termination of many algorithms. Under some assumptions upper bounds for the computational complexity of such algorithms can be extracted by analyzing the length of controlled bad sequences.

We obtain tight upper bounds, in terms of the Fast Growing Hierarchy, for the length of controlled decreasing sequences of multisets over the natural multiset ordering. Then we study the majoring wqo $\leq_{\mathrm{maj}}$ of sets of tuples, which informally states that $A \leq_{\mathrm{maj}} B$ iff every element of A is majorized by an element of B. We linearize this wqo into the multiset well-order, to obtain an upper bound for the length of controlled $\leq_{\mathrm{maj}}$-bad sequences. We finally apply this result to upper-bound the computational complexity of the emptiness problem for ATRA, a class of automata over data trees.

Keywords: Well quasi order, Majoring ordering, Multiset ordering, Tree automata, Fast Growing Hierarchy.

ABSTRACT EXTENDIDO

Un cuasi-orden o preorden es una relación binaria $\leq$ sobre un conjunto dado A que es reflexiva y transitiva. Una secuencia $\mathbf{x} = x_0, x_1, x_2, \ldots$ de elementos de A se dice *buena* si existen $i < j$ tal que $x_i \leq x_j$. Una secuencia es *mala* si no es buena. Un *buen cuasi-orden* (wqo, por sus siglas en inglés) es un cuasi-orden en donde todas las secuencias infinitas son buenas, o, equivalentemente, todas las secuencias malas son finitas.

En principio, una secuencia mala sobre un wqo puede ser arbitrariamente larga. Tomemos, por ejemplo, el orden producto entre n-uplas de números naturales, definido de la siguiente manera: si $\langle x_1, \ldots, x_n\rangle, \langle y_1, \ldots, y_n\rangle \in \mathbb{N}^n$ entonces

$$\langle x_1, \ldots, x_n\rangle \leq_{\mathrm{pr}} \langle y_1, \ldots, y_n\rangle \overset{\mathrm{def}}{\Leftrightarrow} (\forall i \in \{1, \ldots, n\})\ x_i \leq y_i.$$

El Lema de Dickson [11], "el teorema matemático más frecuentemente redescubierto" según [5], dice que $(\mathbb{N}^n, \leq_{\mathrm{pr}})$ es un wqo. Es fácil verificar que para cualquier $N \in \mathbb{N}$ la secuencia

$$\mathbf{x} = \langle 0,1\rangle, \langle N,0\rangle, \langle N-1,0\rangle, \langle N-2,0\rangle, \ldots, \langle 1,0\rangle, \langle 0,0\rangle \qquad (\star)$$

es mala. Así, en general no hay una cota para la longitud de las secuencias malas que empiezan con un cierto elemento dado: las secuencias malas son finitas, pero pueden ser *arbitrariamente* largas.

Si ahora restringimos de alguna manera el 'tamaño' que pueden tener los elementos en la secuencia, la situación cambia. Una *función de norma* $|\cdot|_A$ sobre un conjunto A es una función $|\cdot|_A : A \to \mathbb{N}$ que provee a cada elemento de A con un entero positivo, su *norma*. La función de norma se dice *propia* si $\{x \in A \mid |x|_A < n\}$ es finito para cada n. Sea $g : \mathbb{N} \to \mathbb{N}$ una función creciente y sea $(A, \leq)$ un wqo con una norma propia. Una secuencia $\mathbf{x} = x_0, x_1, x_2 \ldots$ se dice *g, t-controlada* si para todo i, $|x_i|_A < g(t+i)$. Decimos que g es la *función de control* para $\mathbf{x}$.

Como una consecuencia del Lema de König, las secuencias malas y controladas sobre wqos no pueden ser *arbitrariamente* largas: dada una función de control g y un t fijo, existe una cota superior para la longitud de las secuencias malas y g, t-controladas.

Volvamos al ejemplo anterior de la secuencia $\leq_{\mathrm{pr}}$-mala de $(\star)$. Supongamos ahora que $\mathbf{x}$ es $g, 0$-controlada, donde fijamos g como la función $g(x) = x + 2$ y $|x|_{\mathbb{N}^2}$ como la norma infinito de x. Se puede verificar que la secuencia mala respecto al orden producto y $g, 0$-controlada *más larga* tiene longitud 8, como lo muestra, por ejemplo, la siguiente secuencia:

$$\langle 1,1\rangle, \langle 2,0\rangle, \langle 1,0\rangle, \langle 0,4\rangle, \langle 0,3\rangle, \langle 0,2\rangle, \langle 0,1\rangle, \langle 0,0\rangle.$$

Dado un wqo $(A, \leq)$, llamemos $L_g^{(A,\leq)}(t)$ a la longitud de las secuencias $\leq$-malas y g, t-controladas más largas de elementos en A. Según el wqo elegido, la función $L_g^{(A,\leq)}$ puede crecer muy rápido, por ejemplo más rápido que cualquier función primitiva recursiva. En esta tesis clasificamos, para ciertos wqos $(A, \leq)$, la función $L_g^{(A,\leq)}$ en términos de su velocidad de crecimiento. Para medir esta velocidad, utilizamos una clasificación de funciones llamada Jerarquía Rápidamente Creciente (*Fast Growing Hierarchy*) ($\mathfrak{F}_\alpha$), introducida por Löb y Wainer [25]. Las clases de esta jerarquía están indexadas por ordinales. Por ejemplo, la clase $\mathfrak{F}_1$ contiene a todas las funciones lineales, la clase $\mathfrak{F}_2$ contiene a todas las funciones elementales, y en general $\bigcup_{n<\omega} \mathfrak{F}_n$ coincide con la clase de las funciones primitivas recursivas. La jerarquía va más allá y clasifica también funciones computables pero no necesariamente primitivas recursivas, como la función de Ackermann, que pertenece a $\mathfrak{F}_\omega$.

La motivación de estudiar las longitudes máximas de las secuencias malas y controladas viene del lado de las pruebas de terminación de algoritmos. A grandes rasgos, en estas demostraciones la idea es que cualquier secuencia de configuraciones sucesivas de un algoritmo $\mathcal{A}$ con cierta entrada k se transforma en una secuencia mala en un cierto wqo. La función de control surge del algoritmo mismo que se está analizando y resulta siempre una función computable. Así, encontrar una cota superior para la longitud de esta secuencia mala se traduce en una cota superior para la complejidad del algoritmo, es decir, para la cantidad de pasos que el algoritmo $\mathcal{A}$ con entrada k necesita para terminar. Cabe aclarar que estos algoritmos resuelven problemas para los que no se conoce una mejor cota de complejidad que la que surge de su análisis de terminación.

En la resolución de restricciones, deducción automática, análisis de programas, sistemas de reescritura, lógica, autómatas y muchos otros campos de la matemática y las ciencias de la computación, los wqos aparecen generalmente bajo la apariencia de herramientas específicas como el lema de Dickson [11] (para tuplas de números naturales), lema de Higman [19] (para palabras y sus subpalabras), el teorema de árboles de Kruskal [23] y su variantes (para los árboles finitos con inmersiones), y recientemente el teorema de Robertson-Seymour [32] (para grafos y sus menores (*minors*)). Existen además muchos (cuasi-) órdenes buenos que se usan para pruebas de terminación, como el orden de multiconjuntos [9], los órdenes mayorante y minorante [13], etc.

En esta tesis estudiaremos la longitud de las secuencias malas y controladas en el orden de multiconjuntos y en el orden mayorante. Utilizaremos estos resultados para establecer cotas a la complejidad del problema del lenguaje vacío (*emptiness problem*) de dos tipos de autómatas sobre árboles con datos.

Orden de multiconjuntos. Un multiconjunto M sobre un conjunto X es una función $X \to \mathbb{N}$. Intuitivamente, un multiconjunto es una generalización de un conjunto en la cual los elementos pueden aparecer repetidos. Si $x \in X$, con $M(x)$ denotamos la multiplicidad de x en M. Sea X un conjunto ordenado por $\leq$ y M, N multiconjuntos finitos sobre X. Definimos:

$$N <_{\mathrm{ms}}^{(\leq)} M \overset{\mathrm{def}}{\Leftrightarrow} M \neq N \wedge (\forall x \in X)[N(x) > M(x) \Rightarrow (\exists y \in X)[y > x \wedge M(y) > N(y)]].$$

Intuitivamente, esto dice que N se puede obtener a partir de M mediante el reemplazo de algunos de sus elementos por un numero finito (posiblemente cero) de elementos menores. Si $(X, \leq)$ es un buen orden, entonces el conjunto de multiconjuntos finitos bajo el orden recién definido también lo es.

Sea $L_{g,n}^{\mathrm{ms}}$ la longitud de la secuencia decreciente y g, t-controlada mas larga de multiconjuntos de tuplas de naturales con el orden lexicográfico. En este trabajo probaremos el siguiente resultado:

Teorema. *Si g es primitiva recursiva y $g(t) \geq t+1$ entonces $L^{\mathrm{ms}}_{g,n}$ tiene una cota superior en el nivel $\mathfrak{F}_{\omega^n}$ de la Jerarquía Rápidamente Creciente. Ademas, esta cota es ajustada.*

Orden mayorante. Sean A y B subconjuntos finitos de $\mathbb{N}^n$, definimos:

$$A \leq^{(\leq_{\mathrm{pr}})}_{\mathrm{maj}} B \overset{\mathrm{def}}{\Leftrightarrow} (\forall x \in A)(\exists y \in B)\ x \leq_{\mathrm{pr}} y.$$

Se sabe que el orden $\leq^{(\leq_{\mathrm{pr}})}_{\mathrm{maj}}$ sobre subconjuntos finitos de $\mathbb{N}^n$ es un wqo y esto es usado en un gran número de resultados de decibilidad. Sea $L^{\mathrm{maj}}_{g,n}$ la longitud de la secuencia mala y g,t-controlada más larga de conjuntos finitos de tuplas de naturales con el orden producto. A partir de linearizar el orden mayorante en el orden de multiconjuntos y aplicar el teorema anterior, probaremos el siguiente resultado:

Teorema. *Si g es primitiva recursiva, entonces $L^{\mathrm{maj}}_{g,n}$ tiene una cota en el nivel $\mathfrak{F}_{\omega^n}$ de la Jerarquía Rápidamente Creciente.*

Aplicaciones. Finalmente, aplicamos la cota obtenida para $L^{\mathrm{maj}}_{g,n}$ para acotar superiormente la complejidad temporal del procedimiento de decisión conocido para el *emptiness problem* —el problema de determinar si el lenguaje aceptado por un autómata es vacío o no— de dos clases de autómatas sobre árboles: ITCA y ATRA. Para estos, ya se conocía una cota inferior no primitiva recursiva.

1. INTRODUCTION

1.1 Well quasi-orders and controlled bad sequences

A quasi-order is a binary relation $\leq$ over a given set A such that is reflexive and transitive. A sequence $\mathbf{x} = x_0, x_1, x_2, \ldots$ of elements of A is called *good* if there are $i < j$ such that $x_i \leq x_j$. A sequence is *bad* if it is not good. A *well quasi-order* (wqo) is a quasi-order where all infinite sequences are good, or, equivalently, all bad sequences are finite.

Wqo's are widely used in termination proofs of algorithms in constraint solving, automated deduction, program analysis, verification and model checking, rewriting systems, logic, etc. They usually appear under the guise of some specific tool like: Dickson's lemma for tuples of natural numbers (see Definition 1.5), Higman's lemma for words and their subwords (see Definition 4.1), Kruskal's tree theorem [23] for homeomorphic embeddings on finite trees and, more recently, Robertson-Seymour theorem [32] for graphs and its minors. However, many other well (quasi-) orders exists which are fundamental for termination arguments, like the multiset ordering [9] or the majoring and minoring orderings on sets [13].

From the analysis of a termination proof of a given algorithm $\mathcal{S}$, whose correctness is grounded in the analysis of certain wqo, one may extract a computational complexity upper bound for $\mathcal{S}$. Roughly, the idea is that any sequence of successive configurations of $\mathcal{S}$ (with respect to a given input) is transformed into a bad sequence in the wqo. Thus, having an upper bound for the length of the bad sequence entails an upper bound for the number of steps that the algorithm needs to terminate.

For example, let us take the subtraction-based version of Euclid's famous algorithm for obtaining the greatest common divisor of two numbers a and b:

Algorithm 1 Euclid's algorithm

```
function GCD(a, b)
    if a = 0 then
        return b
    while b ≠ 0 do
        if a > b then
            a ← a − b
        else
            b ← b − a
    return a
```

Let a_i and b_i be the value of the variables a and b in the i-th iteration of the algorithm's cycle. For input a and b a *run* of Algorithm 1 is represented by a sequence

$$\langle a_0, b_0\rangle, \langle a_1, b_1\rangle, \langle a_2, b_2\rangle, \ldots, \tag{1.1}$$

where $a_0 = a$ and $b_0 = b$. The run terminates if and only of the sequence is finite. For example, the run of GCD(9,7) is:

$$\langle 9,7\rangle, \langle 2,7\rangle, \langle 2,5\rangle, \langle 2,3\rangle, \langle 2,1\rangle, \langle 1,1\rangle, \langle 1,0\rangle. \tag{1.2}$$

One way to prove termination, is to find a function $f : \mathbb{N}^2 \to \mathbb{N}$ which converts a run like (1.1) into a *decreasing*, hence finite, sequence of natural numbers:

$$f(a_0, b_0) > f(a_1, b_1) > f(a_2, b_2), \ldots. \tag{1.3}$$

In the example of Algorithm 1, we may take $f(x, y) = x + y$. So for the run of (1.2) we get

$$16, 9, 7, 5, 3, 2, 1.$$

In general a *run* of a given algorithm $\mathcal{S}$ with certain input x is a sequence

$$S_0, S_1, S_2, \ldots \tag{1.4}$$

such that S_i is a tuple containing the values of all the variables of $\mathcal{S}$ at step i. Each S_i is a *configuration* of $\mathcal{S}$ on input x. We can prove termination of $\mathcal{S}$ on input x if there is a function f which maps each configuration into a natural number and such that successive configurations S_i and S_{i+1} of a run of $\mathcal{S}$ satisfies $f(S_i) > f(S_{i+1})$. In fact, this last condition may be too strong in some cases. It suffices that for each run (1.4) the sequence $f(S_1), f(S_2), f(S_3), \ldots$ be a bad sequence over some wqo —which would typically depend on the nature of $\mathcal{S}$.

Let us consider a more complex example. Suppose the following non-deterministic algorithm extracted from [34]:

Algorithm 2 Very fast growing algorithm

```
function SIMPLE(a, b)
    c ← 1
    While a > 0 ∧ b > 0
            ⟨a, b, c⟩ ← ⟨a − 1, b, 2c⟩
        Or
            ⟨a, b, c⟩ ← ⟨2c, b − 1, 1⟩
    end
```

The *product ordering* $\leq_{\text{pr}}$ is defined over $\mathbb{N}^k$ as follows:

$$\langle x_1, x_2 \ldots, x_k\rangle \leq_{\text{pr}} \langle y_1, y_2, \ldots, y_k\rangle \overset{\text{def}}{\Leftrightarrow} (\forall i \in \{1, \ldots, k\})\ x_i \leq y_i \tag{1.5}$$

Dickson's Lemma [11], "The most frequently rediscovered mathematical theorem" according to [5], is the statement that $(\mathbb{N}^k, \leq_{\text{pr}})$ is a wqo.

No matter the choice made in each iteration, it can be shown that in any run

$$\langle a_0, b_0, c_0\rangle, \langle a_1, b_1, c_1\rangle, \langle a_2, b_2, c_2\rangle, \ldots$$

it occurs that $\langle a_i, b_i\rangle \not\leq_{\text{pr}} \langle a_j, b_j\rangle$ for all $j > i$, i.e., any run is a bad sequence in $(\mathbb{N}^2, \leq_{\text{pr}})$. Hence, the algorithm terminates for any input a and b. Now, how many steps can this algorithm take to terminate? Consider one possible sequence of computation for SIMPLE(3,3):

$$\langle 3,3,2^0\rangle, \langle 2,3,2^1\rangle, \langle 1,3,2^2\rangle, \langle 2^3,2,1\rangle, \ldots, \langle 1,2,2^{2^3-1}\rangle, \langle 2^{2^3},1,1\rangle, \ldots, \langle 1,1,2^{2^{2^3}-1}\rangle, \langle 0,1,2^{2^{2^3}}\rangle$$

This run has length $3 + 2^3 + 2^{2^3} + 1$, which is non-elementary[1] in the size of the initial values (that is to say, for instance, that it cannot even be computed in EXPSPACE). With the tools developed in [14] for the study of complexity bounds for termination proofs that relies on Dickson's Lemma, this non-elementary lower bound is proven to be tight.

We have seen that a run of GCD can be mapped to a decreasing sequence of $\mathbb{N}$. Notice that it can also be trivially mapped to a bad sequence of $\mathbb{N}^2$ over the product ordering. Observe that to count the number of steps that GCD needs to terminate, it is better to use the former approach over the latter, since it leads to sharper upper bounds.

We aim at finding upper bounds for the length of bad sequences over some wqos. But with no further hypothesis this task is simply impossible, because bad sequences can be, in principle, *arbitrarily* large. For example, for $\mathbb{N}^2$ and any $N \in \mathbb{N}$, the sequence

$$\langle 0,1\rangle, \langle N,0\rangle, \langle N-1,0\rangle, \langle N-2,0\rangle, \dots, \langle 1,0\rangle, \langle 0,0\rangle \tag{1.6}$$

is bad and has length greater than N. What makes this possible is the *uncontrolled* jump from an element like $\langle 0,1\rangle$ to an *arbitrarily* large next element $\langle N,0\rangle$. Therefore, in general there is no bound to the length of a bad sequence starting with a given element.

In practice, in the analysis of termination proofs, one has two additional assumptions of a wqo $(A, \leq)$. First, one has some effective way of measuring the *size* of each element $x \in A$, notated $|x|_A$ or simply $|x|$.

Definition 1.1.1. [33] A *norm function* $|\cdot|_A$ over a set A is a mapping $|\cdot|_A : A \to \mathbb{N}$ that provides every element of A with a positive integer, its *norm*. The norm function is said to be *proper* if $\{x \in A \mid |x|_A < n\}$ is finite for every n.

Second, we may restrict ourselves to bad sequences $\mathbf{x} = x_0, x_1, x_2 \dots$ with a *controlled behaviour*, which means that there is an effective way of computing, given i, an upper bound for $|x_i|$.

Definition 1.1.2. Let $g : \mathbb{N} \to \mathbb{N}$ be a computable increasing function and let $(A, \leq)$ be a wqo with a proper norm. A bad sequence $\mathbf{x} = x_0, x_1, x_2 \dots$ is g, t*-controlled* if for all i, x_i is g, t-controlled, i.e. $|x_i|_A < g(t + i)$. We say that g is the *control function* for $\mathbf{x}$.

As a consequence of König's Lemma, controlled bad sequences over wqos cannot be *arbitrarily* large: given a control, there exist upper bounds for their lengths.

Let us go back to the example of the $\leq_{\text{pr}}$-bad sequence in (1.6). If we further impose that the sequence is $g, 0$-controlled, where we fix g to be $g(x) = x + 2$ and $|x|_{\mathbb{N}^2}$ to be the infinity norm (see (3.1)) of x then the reader may verify that the *longest* $g, 0$-controlled bad sequence has length 8, as shown by the following sequence:

$$\langle 1,1\rangle, \langle 2,0\rangle, \langle 1,0\rangle, \langle 0,4\rangle, \langle 0,3\rangle, \langle 0,2\rangle, \langle 0,1\rangle, \langle 0,0\rangle. \tag{1.7}$$

In this thesis we give upper bounds for the length of g, t-controlled bad sequences, when t is a parameter. That is, given a well (quasi) order under study (we address *multiset* and *majoring*, to be defined later on) $(A, \leq)$, we define $L_g^A(t)$ as the length of the longest g, t-controlled bad sequence in $(A, \leq)$, and we study upper bounds for L_g^A, which are classified in the Fast Growing Hierarchy $(\mathfrak{F}_\alpha)_{\alpha < \epsilon_0}$ of Löb and Wainer [25], a sub-recursive hierarchy which classifies functions according to its rate of growth.

[1] The class $\mathcal{E}$ of elementary functions is the smallest class of functions containing the initial functions $x + y$ and $x - y$, closed under composition and closed under bounded sum and product.

1.2 History and related work

The theory of well quasi-orderings was initially developed by Higman [19] (under the name of "finite basis property") and by Erdös and Rado in an unpublished manuscript. Some early evidence of the theory, however, had already appeared in a work by Neumann [28]. Further developments were [31, 22, 23], and more recently [32].

Wqo's have become a key ingredient in a great number of decidability/ finiteness/regularity results appearing in areas like termination proofs for rewriting systems [8, 10], their extensions [30, 24], complexity upper bounds [12, 27], well-structured transition systems [2, 16, 1, 3], etc. Much more recent is the study of *constructive* aspects of the theory of wqo's.

For the wqo $(\mathbb{N}^n, \leq_{\text{pr}})$ (Dickson's lemma), McAloon [26], in 1984, shows an upper bound for the length of the longest g, t-controlled bad sequence for linear g and place it in the level $\mathfrak{F}_{n+1}$ of the Fast Growing Hierarchy. In 1986, Clote [7] simplifies McAloon's argument but gives a upper bound which is not as tight: $\mathfrak{F}_{n+6}$. All of these proofs are rather complex, and none of them is self-contained. In 2011, D. and S. Figueira, Schmitz and Schnoebelen [14] show an upper bound in $\mathfrak{F}_n$ with a much simpler argument based on a more general mathematical framework of disjoint unions of powers of $\mathbb{N}$. In 2012, Abriola, Figueira y Senno [4] provide an even simpler argument based in the idea of linearizing the wqo $(\mathbb{N}, \leq_{\text{pr}})$ to $\mathbb{N}$ with the lexicographic ordering, which is a well order. This result is based on a constructive proof of Dickson's lemma given by Harwood, Moller y Setzer [18].

For the wqo Γ_p^* (finite words over a finite alphabet with p symbols) with the subword ordering (Higman's Lemma), in 1998, Cichoń and Tahhan Bittar exhibit a reduction method, deducing bounds (for tuples of) words on Γ_p^* from bounds on the Γ_{p-1}^* case. Their decomposition is clear and self-contained, with the control function made explicit. The paper ends up with some in-equalities [6, §8], from which it is not clear what precisely are the upper bounds one can extract. In 2002, Touzet claims a bound on $\mathfrak{F}_{\omega^p}$ with an analysis based on iterated residuations but the proof (given in [35]) is incomplete. In [36, Corollary 6.3], Weiermann gives an $\mathfrak{F}_{\omega^{p-1}}$-like bound for Γ_p^* for sequences produced by term rewriting systems, but his analysis is considerably more involved (as can be expected since it applies to the more general Kruskal Theorem). In 2011, based on the techniques developed in [14], Schnoebelen and Schmitz [33] exhibit a new and self-contained proof of a result which is even more general than Weiermann's. Finally these authors, in [17], extend their results from [33] to work with words over infinite alphabets and an upper bound in $\mathfrak{F}_{\omega^{\omega^k}}$ is given for the length of the longest bad sequence in $(\mathbb{N}^k)^*$ with the subword ordering.

1.3 Main contribution and outline

In [21] Jurdziński and Lazić introduce the class ATRA (alternating top-down register automata) of automata over data trees with alternating control and one register to store and test data. They prove that the emptiness problem for ATRA —i.e., the problem of determining whether the language that an automaton of such class accepts is the empty one— is decidable. This is done by first proving decidability of the emptiness problem for ITCA (incrementing tree counter automata, also introduced in [21]) and then giving a PSPACE translation of ATRA automata to ITCA automata. Termination of such decision procedure relies on the *majoring* wqo, which we will define later on. Along with the decision procedure, a non primitive recursive lower bound on the computational complexity is established.

Our main contribution is an upper bound for the complexity of the emptiness problem for ITCA and ATRA, which we will reach with the following plan:

- In Chapter 2, we start with some general definitions of the theory of well-quasi-orders and introduce the reader to the Fast Growing Hierarchy $(\mathfrak{F}_\alpha)_{\alpha<\epsilon_0}$.
- In Chapter 3, we give tight upper bounds, in terms of the Fast Growing Hierarchy, for $L^{\mathrm{ms}}_{g,n}(t)$, the length of the longest g, t-controlled decreasing sequence of multisets of $\mathbb{N}^n$ with respect to the multiset ordering and the underlying lexicographic ordering.
- In Chapter 4, we show that one can linearize the majoring ordering to the multiset ordering, thus obtaining an upper bound for $L^{\mathrm{maj}}_{g,n}(t)$, the length of the longest g, t-controlled $\leq_{\mathrm{maj}}$-bad sequence of finite subsets of $\mathbb{N}^n$.
- Finally, in Chapter 5, we arrive to the desired upper bound for the complexity of the mentioned emptiness problems by showing a translation from a sequence of configurations of ITCA into a sequence of finite subsets of tuples which is bad with respect to the majoring ordering.

Most of the results of this thesis were published in [4].

2. PRELIMINARIES

2.1 Notation

If A is a set then $|A|$ denotes its cardinality. If $x \in A^n$ then the i-th coordinate of x is denoted $x[i]$, So $x = \langle x[1], \ldots, x[n] \rangle$. Sequences are always in boldface and if $\mathbf{x}$ is a finite sequence then $|\mathbf{x}|$ denotes its length. The concatenation of the sequence $\mathbf{x}$ and the element x at the rightmost place is denoted $\mathbf{x}^\frown x$.

2.2 The Fast Growing Hierarchy

The Fast Growing Hierarchy $(\mathfrak{F}_\alpha)_{\alpha<\epsilon_0}$ of Löb and Wainer [25] is a way of characterizing subrecursive functions by their rate of growth. This classification comprises both a hierarchy within the primitive recursive functions, and generalizations of it like Ackermann function, etc.

To give its formal definition we first need to present some definitions regarding countable ordinal numbers.

Ordinal terms. Let ω be the first infinite ordinal, i.e., the order type of $\mathbb{N}$ with the usual ordering. We will be considering countable ordinals in Cantor Normal Form (CNF), i.e., ordinals of the form:

$$\alpha = \omega^{\beta_1} + \omega^{\beta_2} + \cdots + \omega^{\beta_m}$$

with $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_m$ (this ordering is defined below) ordinals in CNF. For example $\alpha = \omega^2 + \omega^1 + \omega^0 + \omega^0 + \omega^0 + \omega^0$ (where $1 = \omega^0, 2 = \omega^0 + \omega^0$)

An ordinal is either 0 (when m=0) or one of the following:

- **A successor ordinal.** If $\beta_m = 0$ then we say that α is a *successor*. $\alpha + 1$ will denote the successor of α.
- **A limit ordinal.** If $\beta_m \neq 0$ then we say that α is a *limit* ordinal. λ will denote a limit ordinal.

The ordering over these ordinals can be defined inductively as follows:

$$\alpha < \alpha' \stackrel{\text{def}}{=} \begin{cases} \alpha = 0 \text{ y } \alpha' \neq 0, \text{ o} \\ \alpha = \omega^\beta + \gamma, \alpha' = \omega^{\beta'} + \gamma' \text{ y} \begin{cases} \beta < \beta', \text{ o} \\ \beta = \beta' \text{ y } \gamma < \gamma' \end{cases} \end{cases}$$

(where $\omega^\beta \geq \gamma, \omega^{\beta'} \geq \gamma'$)

The Fast Growing Hierarchy $(F_\alpha)_{\alpha<\epsilon_0}$**.** It is defined as

$$\begin{array}{rcl} F_0(x) & \stackrel{\text{def}}{=} & x+1 \\ F_{\alpha+1}(x) & \stackrel{\text{def}}{=} & F_\alpha^{x+1}(x) \\ F_\lambda & \stackrel{\text{def}}{=} & F_{\lambda_x}(x), \end{array}$$

where in general g^k denotes the k-th iteration of g (i.e. $g^1 = g$ and $g^{k+1} = g \circ g^k$), α is an ordinal, $\epsilon_0 = \sup\{\omega, \omega^\omega, \omega^{\omega^\omega}, \dots\}$, $\lambda < \epsilon_0$ is a limit ordinal and $(\lambda_x)_{x\leq\omega}$ is an increasing sequence of ordinals with limit λ (a *fundamental sequence*), which we fix to be:

$$(\gamma + \omega^{\beta+1})_x \stackrel{\text{def}}{=} \gamma + \omega^\beta \cdot (x+1) \qquad (\gamma + \omega^\lambda)_x \stackrel{\text{def}}{=} \gamma + \omega^{\lambda_x}.$$

The class $\mathfrak{F}_\alpha$ of the Fast Growing Hierarchy is the closure under substitution and limited recursion (defined below) of the constant, sum, projections, and the functions F_β with $\beta \leq \alpha$.

- ***substitution*** If $h_0, h_1, \dots, h_n \in \mathfrak{F}_\alpha$, then $f \in \mathfrak{F}_\alpha$, where

$$f(x_1, \dots, x_n) = h_0(h_1(x_1, \dots, x_n), \dots, h_n(x_1, \dots, x_n))$$

- ***limited recursion*** If $h_1, h_2, h_3 \in \mathfrak{F}_\alpha$, then $f \in \mathfrak{F}_\alpha$, where

$$\begin{array}{rcl} f(0, x_1, \dots, x_n) & = & h_1(x_1, \dots, x_n) \\ f(y+1, x_1, \dots, x_n) & = & h_2(y, x_1, \dots, x_n, f(y, x_1, \dots, x_n)) \\ f(y, x_1, \dots, x_n) & \leq & h_3(y, x_1, \dots, x_n) \end{array}$$

It can be shown that $\mathfrak{F}_0 = \mathfrak{F}_1$ contains all linear functions, $\mathfrak{F}_2$ contains all the elementary functions, $\mathfrak{F}_3$ contains al the tetration functions. $\bigcup_{n<\omega} \mathfrak{F}_n$ is the class of all primitive recursive functions. In general, $\bigcup_{\alpha<\omega^k} \mathfrak{F}_\alpha$ is the class of k-recursive functions [29], i.e. the class of functions obtainable from the initial functions by composition and nested recursion in at most k variables. Nested recursion in, at least, 2 variables escapes from the class of primitive recursive functions. For example, the following function ϕ defined from $n+1$ by nested recursion in two variables is not primitive recursive [29, §10.1]:

$$\begin{array}{rcl} \phi(0, n) & = & n+1 \\ \phi(m+1, 0) & = & \phi(m, 1) \\ \phi(m+1, n+1) & = & \phi(m, \phi(m+1, n)) \end{array}$$

There are a number of important *monotonicity* results regarding the Fast Growing Hierarchy: for ordinals $\alpha < \beta < \epsilon_0$, the function F_α is strictly increasing, $F_{\alpha+1} \geq F_\alpha$, F_α is eventually majorized by F_β, and then $\mathfrak{F}_\alpha \subsetneq \mathfrak{F}_\beta$ (except $\alpha = 0$ and $\beta = 1$), etc. For more results on the Fast Growing Hierarchy, cf. [25].

3. MULTISET ORDERING

In this chapter we give tight upper bounds, in terms of the Fast Growing Hierarchy, for the length of the longest g,t-controlled decreasing sequence of multisets of $\mathbb{N}^n$ with respect to the multiset ordering and the underlying lexicographic ordering. We first formally introduce the orderings involved (multiset and lexicographic), then we study the maximizing strategy to get the longest decreasing sequence of controlled multiset, and finally we show lower and upper bounds for the length of such sequence.

A multiset M over a set X is a function $X \to \mathbb{N}$. Intuitively a multiset is a generalization of a set, where elements may be repeated. For $x \in X$, $M(x)$ is called the *multiplicity* of x. A multiset is finite if the set of elements with positive multiplicity is finite. We notate $x \in M$ for $M(x) > 0$. Let $\mathcal{M}_{<\infty}(X)$ denote the class of finite multisets over X.

Let $(X, \leq)$ be a poset (i.e., $\leq$ is a quasi-order which is also antisymmetric) and let $M, N \in \mathcal{M}_{<\infty}(X)$. The *multiset ordering* is defined as follows:

$$N <_{\text{ms}}^{(\leq)} M \stackrel{\text{def}}{\Leftrightarrow} M \neq N \wedge (\forall x \in X)[N(x) > M(x) \Rightarrow (\exists y \in X)[y > x \wedge M(y) > N(y)]].$$

Intuitively, it says that N can be obtained from M by replacing some elements by finitely many (possibly zero) smaller (with respect to $\leq$) elements. If $(X, \leq)$ is a well-order then $(\mathcal{M}_{<\infty}(X), \leq_{\text{ms}}^{(\leq)})$ also is.

The *lexicographic ordering* over $\mathbb{N}^n$ is defined as follows:

$$x <_{\text{lex}} y \stackrel{\text{def}}{\Leftrightarrow} x[1] < y[1] \vee (x[1] = y[1] \wedge \langle x[2], \ldots, x[n] \rangle <_{\text{lex}} \langle y[2], \ldots, y[n] \rangle) .$$

For $x \in \mathbb{N}^n$, recall that the *infinity norm* is defined as

$$|x|_\infty \stackrel{\text{def}}{=} \max\{x[1], \ldots, x[n]\}. \tag{3.1}$$

We will study $(\mathcal{M}_{<\infty}(\mathbb{N}^n), \leq_{\text{ms}}^{(\leq_{\text{lex}})})$, the multiset ordering of finite multisets of tuples with the underlying lexicographic ordering. In this context, we write $\leq_{\text{ms}}$ for $\leq_{\text{ms}}^{(\leq_{\text{lex}})}$. Observe that it is a well-order because $(\mathbb{N}^n, \leq_{\text{lex}})$ is so. We need a notion of *g,t-controlled sequence of (multi)sets*. By Definition 1.1.2 it suffices to give a proper norm:

Definition 3.0.1 (A proper norm of sets and multisets of tuples)**.** Given $X \in \mathcal{M}_{<\infty}(\mathbb{N}^n)$, we define $|X|$, the norm of X, as the maximum between $\max_{x \in \mathbb{N}^n} X(x)$ and $\max\{|x|_\infty \mid x \in \mathbb{N}^n \wedge X(x) > 0\}$. For $X \in \mathcal{P}_{<\infty}(\mathbb{N}^n)$, $|X|$ is defined analogously, as any set is a multiset.

We denote by $L_{g,n}^{\text{ms}}(t)$ the length of the longest g,t-controlled decreasing sequence in $(\mathcal{M}_{<\infty}(\mathbb{N}^n), \leq_{\text{ms}}^{(\leq_{\text{lex}})})$, i.e. a sequence of finite multisets of $\mathbb{N}^n$, with the underlying lexicographic ordering.

3.1 Maximizing strategy

To study the longest g, t-controlled $\leq_{\text{ms}}$-descending sequence of multisets we define the maximizing strategy which, given a nonempty g, t-controlled multiset M, determines the greatest $g, (t+1)$-controlled multiset N which is smaller than M. The strategy says that to obtain N one should take out one of the minimum elements of M, say m, (i.e. decrement in one the multiplicity of m) and add as many elements smaller than m as the control function permits.

For the rest of this section, assume $(X, \leq)$ is a well-order. We write $<_{\text{ms}}$ instead of $<_{\text{ms}}^{(\leq)}$. Let $M \in \mathcal{M}_{<\infty}(X)$ which is g, t-controlled and a proper norm $|\cdot|_X = |\cdot|$ for X. We define the g, t-predecessor of M as follows: For $x \in X$,

$$\text{PRED}_t^g(M)(x) \stackrel{\text{def}}{=} \begin{cases} g(t+1) - 1 & x < \min M \wedge |x| < g(t+1); \\ M(x) - 1 & x = \min M; \\ M(x) & \text{otherwise.} \end{cases}$$

where $\min M \stackrel{\text{def}}{=} \min\{x \mid M(x) > 0\}$.

Lemma 3.1.1. *Let M be a nonempty finite multiset over a totally ordered set P, which is g, t-controlled and let $N = \text{PRED}_t^g(M)$. Then (1) N is $g, (t+1)$-controlled; (2) $N <_{\text{ms}} M$; and (3) if N' is $g, (t+1)$-controlled and $N' <_{\text{ms}} M$ then $N' \leq_{\text{ms}} N$.*

Proof. (1) It is clear from the definition of N and the fact that g is monotone increasing.

(2) It is obvious that $M \neq N$. By definition, if $N(x) > M(x)$ then $x < m = \min M$ and $M(m) > N(m)$.

(3) Assume $N' < M$ is $g, (t+1)$-controlled. We show that if $N'(x) > N(x)$ then there is $z > x$ such that $N(z) > N'(z)$. Suppose $N'(x) > N(x)$.

- Suppose $x < \min M$. Then $N(x) = g(t+1) - 1 \geq N'(x)$, contradicting $N'(x) > N(x)$.
- Suppose $x > \min M$. Then $N(x) = M(x)$ and therefore $N'(x) > M(x)$. Since $N' <_{\text{ms}} M$ there is $z > x$ such that $N(z) = M(z) > N'(z)$.
- Suppose $x = \min M$. Then $N(x) = M(x) - 1$, and so $N'(x) \geq M(x)$. If $N'(x) > M(x)$ then, since $M <_{\text{ms}} N'$, there is $z > x$ with $M(z) > N'(z)$. For such z, by definition of N, we have $N(z) = M(z) > N'(z)$. If $N'(x) = M(x)$ then, since $N' \neq M$, there is y such that $N'(y) \neq M(y)$. Any such y must be different from x. Suppose that all such y's were smaller than $x = \min M$. In this case $M \leq_{\text{ms}} N'$ and this contradicts the hypothesis. Hence there is $y > x$ such that $N'(y) \neq M(y)$. If $N'(y) > M(y)$, there is $z > y > x$ such that $N'(z) < M(z) = N(z)$. If $N'(y) < M(y)$, since $M(y) = N(y)$, we conclude $N'(y) < N(y)$.

□

We represent a finite multiset M such that $\{x \mid M(x) > 0\} = \{x_1, \ldots, x_n\}$ as

$$M \stackrel{\text{def}}{=} M(x_1) \cdot x_1 + \cdots + M(x_n) \cdot x_n.$$

For a finite multiset M, let $L_{g,M}(t)$ denote the length minus one of the longest g, t-controlled and $<_{\text{ms}}$-decreasing sequence of multisets starting with the multiset M. For $x \in X$, let $o_{g,x}(t) = t + L_{g,1\cdot\{x\}}(t)$.

Lemma 3.1.2. *If $k \geq 1$ then $L_{g,k\cdot\{x\}}(t) = \sum_{i=0}^{k-1} L_{g,1\cdot\{x\}}(o^i_{g,x}(t))$.*

Proof. We write L_k for $L_{g,k\cdot\{x\}}$ and o for $o_{g,x}$. We first show the following

Fact 3.1.3. $o^i(t) = t + \sum_{j=0}^{i-1} L_1(o^j(t))$.

Proof. By induction in $i \geq 0$. If $i = 0$ it is trivial. Now

$$\begin{aligned}
o^{i+1}(t) &= o(o^i(t)) \\
&= o(t + \sum_{j=1}^{i-1} L_1(o^j(t))) && \text{(ind. hyp.)} \\
&= t + \sum_{j=1}^{i-1} L_1(o^j(t))) + L_1(o^i(t)) && \text{(Definition of } o \text{ and ind. hyp.)} \\
&= t + \sum_{j=1}^{i} L_1(o^j(t)).
\end{aligned}$$

□

Now we show the statement of the Lemma by induction in $k \geq 1$. If $k = 1$ it is straightforward. Now suppose that the longest g,t-controlled decreasing sequence of multisets beginning with $(k+1)\cdot\{x\}$ is

$$M_1 >_{\text{ms}} M_2 >_{\text{ms}} \ldots, >_{\text{ms}} M_{l_1} >_{\text{ms}} N_2 >_{\text{ms}} N_3 >_{\text{ms}} \ldots >_{\text{ms}} N_{l_2}$$

of length $l_1 + l_2 - 1$ and where $M_1 = (k+1)\cdot\{x\}$, $l_1 = L_k(t) + 1$, $M_{l_1} = 1\cdot\{x\}$, $l_2 = L_1(t + L_k(t)) + 1$ and $N_{l_2} = \emptyset$. We have

$$\begin{aligned}
L_{k+1}(t) &= l_1 + l_2 - 2 \\
&= L_k(t) + L_1(t + L_k) \\
&= \sum_{i=0}^{k-1} L_1(o^i(t)) + L_1(t + \sum_{i=0}^{k-1} L_1(o^i(t))) && \text{(ind. hyp.)} \\
&= \sum_{i=0}^{k-1} L_1(o^i(t)) + L_1(o^k(t)) + 1 && \text{(Fact 3.1.3)} \\
&= \sum_{i=0}^{k} L_1(o^i(t)),
\end{aligned}$$

and this concludes the proof. □

Corollary 3.1.4. *For $k \geq 1$, $L_{g,k\cdot\{x\}} \geq L^k_{g,1\cdot\{x\}}$.*

Corollary 3.1.5. *For $k \geq 1$, $L_{g,k\cdot\{x\}}(t) \leq k \cdot L_{g,1\cdot\{x\}}(o^{k-1}_{g,x}(t))$.*

In the sequel we fix $(X, \leq)$ to be $(\mathbb{N}^n, \leq_{\text{lex}})$. If $M \in \mathcal{M}_{<\infty}(\mathbb{N}^n)$ then $P_{g,n}(M,t)$ denotes the length minus one of the longest g,t-controlled $<_{\text{ms}}$-decreasing sequence of multisets starting with M. If M consists of one copy of $(x_1, \ldots, x_n)$, we simply write $P_{g,n}(x_1, \ldots, x_n, t)$ instead of $P_{g,n}(1\cdot\{(x_1, \ldots, x_n)\}, t)$. Observe that, having fixed $(X, \leq)$, we have $L_{g,M}(t) = P_{g,n}(M,t)$.

3.2 Lower bound

Define $G_{g,n} : \mathbb{N}^{n+1} \setminus \{(0,\dots,0)\} \to \mathbb{N}$ by multiple recursion as:

$$G_{g,n}(0,\dots,0,1,t) \stackrel{\text{def}}{=} g(t+1) \tag{3.2}$$

$$G_{g,n}(\overline{x},x_n+1,t) \stackrel{\text{def}}{=} G_{g,n}^{g(t+1)-1}(\overline{x},x_n,t),\ \text{for } \overline{x}=x_1,\dots,x_{n-1} \tag{3.3}$$

$$G_{g,n}(\overline{x},x_j+1,\overline{0},t) \stackrel{\text{def}}{=} G_{g,n}(\overline{x},x_j,g(t+1)-1,\overline{0},t),\ \text{for } \overline{x}=x_1,\dots,x_{j-1} \tag{3.4}$$

Equation (3.3) applies when $x_i > 0$ for some i, and (3.4) when $j < n$. $G^k_{g,n}(\overline{a},b)$ denotes the k-th iteration of $G_{g,n}$ in the last component, i.e. $G^1_{g,n}(\overline{a},b) = G_{g,n}(\overline{a},b)$ and $G^{k+1}_{g,n}(\overline{a},b) = G_{g,n}(\overline{a},G^k_{g,n}(\overline{a},b))$.

Lemma 3.2.1. *If $g(x) \geq x+1$ then $P_{g,n} \geq G_{g,n}$.*

Proof. By induction in the lexicographic order of $x_1,\dots,x_n$. If $(x_1,\dots,x_{n-1},x_n) = (0,\dots,0,1)$ then the longest g,t-controlled $<_{\text{ms}}$-decreasing sequence starting with $1\cdot\{(0,\dots,0,1)\}$ is

$$1\cdot\{(0,\dots,0,1)\} >_{\text{ms}} (g(t+1)-1)\cdot\{(0,\dots,0,0)\} >_{\text{ms}}$$
$$>_{\text{ms}} (g(t+1)-2)\cdot\{(0,\dots,0,0)\} >_{\text{ms}} \dots >_{\text{ms}} 0\cdot\{(0,\dots,0,0)\}$$

which has length $g(t+1)+1$ and then

$$\begin{aligned} P_{g,n}(0,\dots,0,1,t) &= g(t+1) \\ &= G_{g,n}(0,\dots,0,1,t). \end{aligned}$$

The longest g,t-controlled $<_{\text{ms}}$-decreasing sequence of multisets starting with

$$1\cdot\{(x_1,\dots,x_{n-1},x_n+1)\}$$

continues as

$$M_1 = \sum\{(g(t+1)-1)\cdot\{(y_1,\dots,y_n)\} \mid (y_1,\dots,y_n) <_{\text{lex}} (x_1,\dots,x_{n-1},x_n+1)\} \tag{3.5}$$

Therefore

$$\begin{aligned} P_{g,n}(x_1,\dots,x_{n-1},x_n+1,t) &\geq P_{g,n}((g(t+1)-1)\cdot\{(x_1,\dots,x_{n-1},x_n)\},t+1) \\ &\geq P^{g(t+1)-1}_{g,n}(x_1,\dots,x_{n-1},x_n,t+1) && \text{(Corollary 3.1.4)} \\ &\geq G^{g(t+1)-1}_{g,n}(x_1,\dots,x_{n-1},x_n,t) && \text{(ind. hyp. and monot. of } G_{g,n}) \\ &= G_{g,n}(x_1,\dots,x_{n-1},x_n+1,t) \end{aligned}$$

The longest g,t-controlled $<_{\text{ms}}$-decreasing sequence of multisets starting with

$$1\cdot\{(x_1,\dots,x_j+1,0,\dots,0)\}$$

has

$$\{(x_1,\dots,x_j,g(t+1),0,\dots,0)\}$$

as one of its terms. Therefore

$$\begin{aligned} P_{g,n}(x_1,\dots,x_j+1,0,0,\dots,0,t) &\geq P_{g,n}(x_1,\dots,x_j,g(t+1)-1,0,\dots,0,t) \\ &\geq G_{g,n}(x_1,\dots,x_j,g(t+1)-1,0,\dots,0,t) && \text{(ind. hyp.)} \\ &= G_{g,n}(x_1,\dots,x_j+1,0,\dots,0,t) \end{aligned}$$

This concludes the proof. □

Theorem 3.2.2. *If $g \geq F_1$ and $g(x) \geq x+2$, then $L^{\mathrm{ms}}_{g,n} \geq F_{\omega^n}$.*

Proof. We show that if $x_i > 0$ for some i then

$$G_{g,n}(x_{n-1},\dots,x_1,x_0,t) \geq F_\alpha(t),$$

where $\alpha = \omega^{n-1}\cdot x_{n-1} + \cdots + \omega^0 \cdot x_0$. We proceed by induction in $(x_{n-1},\dots,x_1,x_0)$. For the base case, observe that $G_{g,n}(0,\dots,0,1,t) = g(t+1) \geq g(t) \geq F_1(t)$. Let $\alpha = \omega^{n-1}\cdot x_{n-1} + \cdots + \omega^0 \cdot x_0$. We have

$$\begin{aligned} G_{g,n}(x_{n-1},\dots,x_1,x_0+1,t) &= G^{g(t+1)-1}_{g,n}(x_{n-1},\dots,x_1,x_0,t) \\ &\geq G^{t+1}_{g,n}(x_{n-1},\dots,x_1,x_0,t) && \text{(monot. of } G_{g,n} \text{ and } g(x) \geq x+1) \\ &\geq F^{t+1}_{\alpha}(t) && \text{(ind. hyp.)} \\ &= F_{\alpha+1}(t) \end{aligned}$$

Let $\beta = \omega^{n-1}\cdot x_{n-1} + \cdots + \omega^j \cdot x_j$ and $j > 0$. We have

$$\begin{aligned} G_{g,n}(x_{n-1},\dots,x_j+1,0,\dots,0,t) &= G_{g,n}(x_{n-1},\dots,x_j,g(t+1)-1,0,\dots,0,t) \\ &\geq G_{g,n}(x_{n-1},\dots,x_j,t+1,0,\dots,0,t) \\ & && \text{(monot. of } G_{g,n} \text{ and } g(x) \geq x+1) \\ &= F_{\beta+\omega^{j-1}\cdot(t+1)}(t) && \text{(ind. hyp.)} \\ &= F_{\beta+\omega^j}(t) \end{aligned}$$

Finally, for all t we have

$$\begin{aligned} L^{\mathrm{ms}}_{g,n}(t) &\geq P_{g,n}(g(t)-1,0,\dots,0,t) \\ &\geq P_{g,n}(t+1,0,\dots,0,t) && \text{(monot. of } P_{g,n} \text{ and } g(x) \geq x+2) \\ &\geq G_{g,n}(t+1,0,\dots,0,t) && \text{(Lemma 3.2.1)} \\ &\geq F_{\omega^{n-1}\cdot(t+1)}(t) \\ &= F_{\omega^n}(t) \end{aligned}$$

and this concludes the proof. □

3.3 Upper bound

Define $U_{g,n} : \mathbb{N}^{n+1} \setminus \{(0,\dots,0)\} \to \mathbb{N}$ by multiple recursion as:

$$U_{g,n}(0,\dots,0,1,t) \stackrel{\text{def}}{=} g(t+1) \tag{3.6}$$

$$U_{g,n}(\overline{x},x_n+1,t) \stackrel{\text{def}}{=} g(t+1)\cdot U_{g,n}(\overline{x},x_n,o^{g(t+1)-1}_{x_1,\dots,x_n}(t+2)) \tag{3.7}$$

$$U_{g,n}(\overline{x},x_j+1,\overline{0},t) \stackrel{\text{def}}{=} U_{g,n}(\overline{x},x_j,g(t+1),\overline{0},t+2) \tag{3.8}$$

where $o_{x_1,\dots,x_n}(t) = t + U_{g,n}(x_1,\dots,x_{n-1},x_n,t)$; equation (3.7) applies when $x_i > 0$ and $\overline{x} = x_1,\dots,x_{n-1}$; and equation (3.8) applies when $j < n$ and $\overline{x} = x_1,\dots,x_{j-1}$.

Lemma 3.3.1. $P_{g,n} \leq U_{g,n}$.

Proof. By induction in the lexicographic order of $x_1, \ldots, x_n$. For (3.6), as in the proof of Lemma 3.2.1, the longest g, t-controlled $<_{\mathrm{ms}}$-decreasing sequence starting with $1 \cdot \{(\overline{0}, 1)\}$ has length $g(t+1)+1$ and then

$$P_{g,n}(\overline{0}, 1, t) = g(t+1) = U_{g,n}(\overline{0}, 1, t).$$

For (3.7) the longest g, t-controlled $<_{\mathrm{ms}}$-decreasing sequence starting with $M_0 = 1 \cdot \{(\overline{x}, x_n+1)\}$ continues with a multiset M_1whose $<_{\mathrm{lex}}$-maximum element is $(\overline{x}, x_n)$, of multiplicity $g(t+1)-1$. Therefore if $N = g(t+1) \cdot \{(\overline{x}, x_n)\}$ then $M_0 >_{\mathrm{ms}} N >_{\mathrm{ms}} M_1$ and N is $g, (t+2)$-controlled. Hence

$$\begin{aligned} P_{g,n}(\overline{x}, x_n+1, t) &\leq P_{g,n}(g(t+1) \cdot \{(\overline{x}, x_n)\}, t+2) \\ &\leq g(t+1) \cdot P_{g,n}(\overline{x}, x_n, \tilde{o}^{g(t+1)-1}_{x_1,\ldots,x_n}(t+2)) \\ &\leq g(t+1) \cdot U_{g,n}(\overline{x}, x_n, o^{g(t+1)-1}_{x_1,\ldots,x_n}(t+2)) = U_{g,n}(\overline{x}, x_n+1, t) \end{aligned}$$

where $\tilde{o}_{x_1,\ldots,x_n}(t) = t + P_{g,n}(x_1, \ldots, x_n, t)$, the second inequality follows from Corollary 3.1.5, and the third one from ind. hyp. and monotonicity of $U_{g,n}$. For (3.7) the longest g, t-controlled $<_{\mathrm{ms}}$-decreasing sequence of multisets starting with $M'_0 = 1 \cdot \{(\overline{x}, x_j+1, \overline{0})\}$ continues with a multiset M'_1 whose $<_{\mathrm{lex}}$-maximum element is $(\overline{x}, x_j, g(t+1)-1, \ldots, g(t+1)-1)$, of multiplicity $g(t+1)-1$. Then $M'_0 >_{\mathrm{ms}} N' >_{\mathrm{ms}} M'_1$, where $N' = 1 \cdot \{(\overline{x}, x_j, g(t+1), \overline{0})\}$, and hence N' is $g, (t+2)$-controlled. Therefore by inductive hypothesis we have

$$\begin{aligned} P_{g,n}(\overline{x}, x_j+1, \overline{0}, t) &\leq P_{g,n}(\overline{x}, x_j, g(t+1), \overline{0}, t+2) \\ &\leq U_{g,n}(\overline{x}, x_j, g(t+1), \overline{0}, t+2) = U_{g,n}(\overline{x}, x_j+1, \overline{0}, t), \end{aligned}$$

and this concludes the proof. □

Theorem 3.3.2. *If g is primitive recursive and $g(t) \geq t+1$ then $L^{\mathrm{ms}}_{g,n}$ has an upper bound in $\mathfrak{F}_{\omega^n}$. Also, this bound is tight.*

Proof. The fact that the bound is tight follows from Theorem 3.2.2. Without loss of generality suppose, $t > 2$ and let $2 \leq e < \omega$ such that $g(t+1) \leq F_e(t)$. By $(\forall^\infty x)\varphi(x)$ we mean that φ holds for almost every x, i.e $(\exists k)(\forall x > k)\varphi(x)$.

Fact 3.3.3. If $x \neq 0$ then $(\forall^\infty t)(\forall x) U_{g,n}(\overline{0}, x, t) \leq F_{3(x-1)+e}(t)$.

Proof. By induction in $x \neq 0$. For $x = 1$, observe that $U_{g,n}(\overline{0}, 1, t) = g(t+1) \leq F_e(t)$. For the inductive step, $o_{\overline{0},x} = t + U_{g,n}(\overline{0}, x, t) \leq t + F_{3(x-1)+e}(t) \leq F_{3(x-1)+e+1}(t)$. Now

$$\begin{aligned} U_{g,n}(\overline{0}, x+1, t) =& g(t+1) \cdot U_{g,n}(\overline{0}, x, o^{g(t+1)-1}_{\overline{0},x}(t+2)) \\ \leq& g(t+1) \cdot F_{3(x-1)+e}(o^{g(t+1)-1}_{\overline{0},x}(t+2)) && \text{(ind. hyp.)} \\ \leq& F_e(t) \cdot F_{p(x)}(F^{g(t+1)-1}_{p(x)+1}(t+2)) && (p(x) \stackrel{\mathrm{def}}{=} 3(x-1)+e) \\ \leq& F_e(t) \cdot F_{p(x)}(F^{g(t+1)+1}_{p(x)+1}(g(t+1)) \\ =& F_e(t) \cdot F_{p(x)}(F_{p(x)+2}(g(t+1)) \\ \leq& F_{p(x)+2}(F_{p(x)+2}(F_{p(x)+2}(F_{p(x)+2}(t))) \\ =& F^4_{p(x)+2}(t) \leq F_{p(x)+3}(t) = F_{3x+e}. && (t \geq 3) \end{aligned}$$

This concludes the proof of the Fact □

Fact 3.3.4. If $x_0 > 0$ then

$$(\forall^\infty t)(\forall x_{n-1},\dots,x_0)[U_{g,n}(x_{n-1},\dots,x_0,t) \le F_\gamma(t) \Rightarrow U_{g,n}(x_{n-1},\dots,x_0+1,t) \le F_{\gamma+3}(t)].$$

Proof. Same idea as in Fact 3.3.3. □

Fact 3.3.5. If $x_i > 0$ for some $i \ge 1$ then $(\forall^\infty t)(\forall \overline{x} = x_{n-1},\dots,x_1) U_{g,n}(\overline{x},0,t) \le F_\alpha(t)$, where $\alpha = x_{n-1}\cdot\omega^{n-1} + x_{n-2}\cdot\omega^{n-2} + \cdots + x_2\cdot\omega^2 + x_1\cdot\omega + 1$.

Proof. By induction in $\overline{x} \ne \overline{0}$. $U_{g,n}(\overline{0},1,0,t) = U_{g,n}(\overline{0},g(t+1),t+2) \le F_{d(t)}(t+2) \le F_{d(t)+1}(d(t)) = F_\omega(d(t)) \le F_{\omega+1}(t)$ where $d(t) \overset{\text{def}}{=} 3(g(t+1)-1)+e$, the first inequality follows from Fact 3.3.3 and the last one is true for all $t \ge k_1$.

Next,

$$\begin{aligned} U_{g,n}(\overline{0},x_1+1,0,t) &= U_{g,n}(\overline{0},x_1,g(t+1),t+2)\\ &\le F_{x_1\cdot\omega+1+r(t)}(t+2)\\ &\le F_{x_1\cdot\omega+1+r(t)}(r(t))\\ &= F_{(x_1+1)\cdot\omega}(r(t))\\ &\le F_{(x_1+1)\cdot\omega+1}(t), \end{aligned}$$

where $r(t) \overset{\text{def}}{=} 3g(t+1)$, the first inequality follows from ind. hyp. and Fact 3.3.4 and the last one is true for all $t \ge k_2 \ge k_1$ (independently of x_1).

Finally, let $\overline{x} = x_{n-1},\dots,x_{j-1}$ and let $\beta = x_{n-1}\cdot\omega^{n-1} + \cdots + x^{j-1}\cdot\omega^{j-1}$.

$$\begin{aligned} U_{g,n}(\overline{x},x_j+1,\overline{0},t) &= U_{g,n}(\overline{x},x_j,g(t+1),\overline{0},t+2)\\ &\le F_{\beta+x_j\cdot\omega^j+g(t+1)\cdot\omega^{j-1}+1}(t+2) && \text{(ind. hyp.)}\\ &\le F_{\beta+x_j\cdot\omega^j+(g(t+1)+1)\cdot\omega^{j-1}}(t+2)\\ &\le F_{\beta+x_j\cdot\omega^j+(g(t+1)+1)\cdot\omega^{j-1}}(g(t+1))\\ &\le F_{\beta+(x_j+1)\cdot\omega^j}(g(t+1)) \le F_{\beta+(x_j+1)\cdot\omega^j+1}(t), \end{aligned}$$

where the last inequality is true for all $t \ge k_3 \ge k_2$ (independently of $\overline{x}, x_j$). □

Now, let t be sufficiently large. If $n = 1$ then

$$\begin{aligned} L^{\text{ms}}_{g,n}(t) &\le P_{g,n}(g(t),t+1)\\ &\le U_{g,n}(g(t),t+1)\\ &\le F_{3(g(t)-1)+e}(t+1)\\ &\le F_{3(g(t)-1)+e+1}(3(g(t)-1)+e) = F_\omega(3(g(t)-1)+e) \in \mathfrak{F}_\omega \end{aligned}$$

where the second inequality follows from Lemma 3.3.1 and the third one from Fact 3.3.3. If $n > 1$ we have:

$$\begin{aligned} L^{\text{ms}}_{g,n}(t) &\le P_{g,n}(g(t),\overline{0},t+1)\\ &\le U_{g,n}(g(t),\overline{0},t+1)\\ &\le F_{g(t)\cdot\omega^{n-1}+1}(t+1)\\ &\le F_{(g(t)+1)\cdot\omega^{n-1}}(g(t))\\ &= F_{\omega^n}(g(t)) \in \mathfrak{F}_{\omega^n}. \end{aligned}$$

The second inequality follows from Lemma 3.3.1 and the third one from Fact 3.3.5. □

4. MAJORING ORDERING

In this chapter we study bad sequences in the majoring ordering of set of tuples and give an upper bound for its length in terms of the Fast Growing Hierarchy.

Definition 4.0.6 (Majoring ordering $\leq_{\text{maj}}$)**.** Let $\mathcal{P}_{<\infty}(X)$ denote the finite and non-empty parts of X. For a wqo $(X, \leq)$ and $A, B \in \mathcal{P}_{<\infty}(X)$, the *majoring ordering* is defined as

$$A \leq_{\text{maj}}^{(\leq)} B \overset{\text{def}}{\Leftrightarrow} (\forall x \in A)(\exists y \in B)\ x \leq y.$$

The fact that the majoring ordering is a wqo follows from Higman's lemma [19], as we will see in Proposition 4.0.7. Let us first introduce Higman's lemma. Given a quasi-order $(X, \leq)$ we define the *subword ordering* over X^* as follows:

$$x_1 \dots x_n \sqsubseteq y_1 \dots y_m \overset{\text{def}}{\Leftrightarrow} (\exists 1 \leq i_1 < \cdots < i_n \leq m)(\forall j \in \{1, \dots, n\})\ x_j \leq y_{i_j}. \tag{4.1}$$

Higman's Lemma is the statement that $(X^*, \sqsubseteq)$ is a wqo.

Proposition 4.0.7. *If $(X, \leq)$ is a wqo, then $(\mathcal{P}_{<\infty}(X), \leq_{\text{maj}}^{(\leq)})$ is a wqo.*

Proof. The fact that this order is reflexive and transitive is immediate from the fact that $(X, \leq)$ is a wqo. The fact of being a wqo is a simple consequence of Higman's Lemma. Each finite set $\{a_1, \dots, a_n\}$ can be seen as a sequence of elements $a_1 \dots a_n$, in any order. In this context, the subword ordering is stricter than the majoring ordering. In other words, if $a_1 \dots a_n \sqsubseteq a'_1 ... a'_m$, then $\{a_1, \dots, a_n\} \leq_{\text{maj}} \{a'_1, \dots, a'_m\}$. By Higman's Lemma the subword ordering over $(X, \leq)$ is a wqo, implying that the majoring ordering is as well. □

For example, the following is a bad sequence in $(\mathcal{P}_{<\infty}(\mathbb{N}^2), \leq_{\text{maj}}^{(\leq_{\text{pr}})})$:

$$\{\langle 3,3\rangle\}, \{\langle 1,4\rangle, \langle 4,1\rangle\}, \{\langle 5,1\rangle, \langle 3,2\rangle\}, \{\langle 2,1\rangle\}, \{\langle 1,5\rangle\} \tag{4.2}$$

We will study $(\mathcal{P}_{<\infty}(\mathbb{N}^n), \leq_{\text{maj}}^{(\leq_{\text{pr}})})$, the majoring ordering of finite sets of tuples with the underlying product ordering. In this context, we write $\leq_{\text{maj}}$ for $\leq_{\text{maj}}^{(\leq_{\text{pr}})}$. Let $L_{n,g}^{\text{maj}}(t)$ denote the length of the longest g, t-controlled bad sequence over $(\mathcal{P}_{<\infty}(\mathbb{N}^n), \leq_{\text{maj}}^{(\leq_{\text{pr}})})$. In the following section we give an upper bound for $L_{n,g}^{\text{maj}}(t)$.

4.1 Simple observations to warm up

Let $L_{n,g}^{\text{pr}}(t)$ be the length of the longest g, t-controlled sequence in the product ordering of Definition 1.5.

Observation 4.1.1. $L_{1,g}^{\text{maj}} = L_{1,g}^{\text{pr}}$.

Proof. If $\mathbf{X} = X_1, \ldots, X_k$ is a bad sequence of finite and non-empty sets of numbers, which is g, t-controlled then $\sup X_1, \ldots, \sup X_k$ is a t-controlled bad sequence of numbers; therefore $L^{\text{maj}}_{1,g} \leq L^{\text{pr}}_{1,g}$. If $\mathbf{x} = x_1, \ldots, x_k$ is a g, t-controlled bad sequence of numbers then $\{x_1\}, \ldots, \{x_k\}$ is a bad sequence of sets of numbers which is also g, t-controlled; hence $L^{\text{pr}}_{1,g} \leq L^{\text{maj}}_{1,g}$. □

Taking sup as in the above proof does not work for $n > 1$. Take $\mathbf{X}$ as in (4.2). $\mathbf{X}$ is a bad sequence but

$$\{\langle 3,3\rangle\}, \{\langle 4,4\rangle\}, \{\langle 5,2\rangle\}, \{\langle 2,1\rangle\}, \{\langle 1,5\rangle\}$$

is not.

Since any bad sequence of n-tuples is trivially a bad sequence of singletons, we have $L^{\text{maj}}_{n,g} \geq L^{\text{pr}}_{n,g}$. It is not difficult to see that equality does not always hold.

Observation 4.1.2. There is g such that $L^{\text{maj}}_{2,g}(0) > L^{\text{pr}}_{2,g}(0)$.

Proof. Take $n = 2$ and control function $g(x) = x + 2$. We saw in the introduction that longest $g, 0$-controlled bad sequence has length 8. However, for example, the following sequence of sets:

$$\{\langle 1,1\rangle\}, \{\langle 0,2\rangle, \langle 2,0\rangle\}, \{\langle 0,2\rangle, \langle 1,0\rangle\}, \{\langle 0,1\rangle, \langle 2,0\rangle\}, \{\langle 0,1\rangle, \langle 1,0\rangle\},$$
$$\{\langle 6,0\rangle\}, \{\langle 5,0\rangle\}, \{\langle 4,0\rangle\}, \{\langle 3,0\rangle\}, \{\langle 2,0\rangle\}, \{\langle 1,0\rangle\};$$

is $g, 0$-controlled, bad and has length 11. □

4.2 Linearizing

The strategy to derive an upper-bound for $L^{\text{maj}}_{n,g}(t)$ will be *linearizing* the wqo $(\mathcal{P}_{<\infty}(\mathbb{N}^n), \leq_{\text{maj}})$ into the well-order $(\mathcal{M}_{<\infty}(\mathbb{N}^n), \leq_{\text{ms}})$ and then use the results of §3.

Let $L^A_g(t)$ denote the length of the longest g, t-controlled bad sequences over the wqo $(A, \leq_A)$ when t is a parameter.

Linearizing the wqo $(A, \leq_A)$ with a proper norm $|\cdot|_A$ into a suitable well linear order $(B, \leq_B)$ with a proper norm $|\cdot|_B$ means finding a function $h : A^+ \to B$ such that for every $\mathbf{a} \in A^+$ and $a \in A$, if $\mathbf{a}^\frown a$ is a bad sequence in $(A, \leq_A)$ then $h(\mathbf{a}) >_B h(\mathbf{a}^\frown a)$. So if $\mathbf{a} = a_0, \ldots, a_k$ is bad in $(A, \leq_A)$ then

$$\mathbf{b} = h(a_0), h(a_0, a_1), h(a_0, a_1, a_2), \ldots, h(\mathbf{a})$$

is descending in $(B, \leq_B)$. Furthermore, for any control function g we seek a control function $\tilde{g}$ such that if $\mathbf{a}$ is g, t-controlled then $|h(\mathbf{a})|_B < \tilde{g}(|\mathbf{a}| + t - 1)$ —here $|\mathbf{a}|$ denotes the length of $\mathbf{a}$.

Hence if $\mathbf{a}$ is g, t-controlled then $\mathbf{b}$ is $\tilde{g}, t$-controlled and therefore from a g, t-controlled bad sequence in $(A, \leq_A)$ one can get a $\tilde{g}, t$-descending sequence in $(B, \leq_B)$ *of the same length.* Hence $L^A_g \leq L^B_{\tilde{g}}$, and the task is now to find an upper bound for $L^B_{\tilde{g}}$. In practice, these upper bounds are easier to devise for well-orders than for wqo's.

4.3 Upper bound

Our linearization will be done in two steps. Given a $\leq_{\text{maj}}$-bad sequence

$$\mathbf{X} = X_0, X_1, \dots X_k$$

of finite and nonempty sets of n-tuples we define an intermediate sequence

$$T_0, T_1, \dots, T_k$$

of trees whose nodes are decorated with n-tuples. From these trees we define a sequence of finite and nonempty multisets of n-tuples

$$\mathbf{M} = M_0, M_1, \dots, M_k$$

We show that if $\mathbf{X}$ is $\leq_{\text{maj}}$-bad then $\mathbf{M}$ is $<_{\text{ms}}$-decreasing. Furthermore, given a control for $\mathbf{X}$, we find a control for $\mathbf{M}$. Using the results of §3 we give an answer to the question of the maximum possible length of a controlled $\leq_{\text{maj}}$-bad sequence of finite sets of n-tuples.

Let $X \subseteq \mathbb{N}^n$. We say X *avoids* x if for all $y \in X$ we have $x \not\leq_{\text{pr}} y$. Since $\mathbf{X}$ is bad, then for any $i < j$, X_j avoids some tuple of X_i. In particular for all $j \in \{1, \dots, k\}$, X_j avoids some tuple of X_0. If a is the $\leq_{\text{pr}}$-supremum of X_0 then $\tilde{\mathbf{X}} = \{a\}, X_1, \dots X_k$ is also a bad sequence. Furthermore, if $\mathbf{X}$ was g, t-controlled then $\tilde{\mathbf{X}}$ also is, and in this case $a \leq_{\text{pr}} \langle g(t)-1, \dots, g(t)-1\rangle$. Even more, if $\mathbf{X}$ is the longest such sequence then $a = \langle g(t)-1, \dots, g(t)-1\rangle$. Therefore, without loss of generality we may assume that all $\leq_{\text{maj}}$-bad sequences of sets analyzed here have a singleton as the first element.

Construction of the trees T_i.

Without loss of generality suppose $X_0 = \{a_0\}$. Define the following sequence of finite trees of n-tuples. By *path* we always refer to a path from the root to a leaf. See Fig. 4.1 for an example of this construction.

- T_0 is a_0, the root.
- T_{i+1} is formed by extending T_i as follows. For any path $a_0, \dots, a_m$ in T_i do the following: if for all $j = 0, \dots, m$, X_{i+1} avoids a_j then add all the elements of X_{i+1} as new children of a_m.

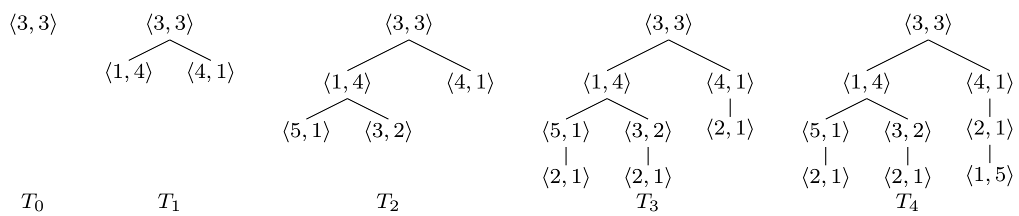


Fig. 4.1: Construction of the trees for the bad sequence X_0, X_1, X_2, X_3, X_4, where $X_0 = \{\langle 3,3\rangle\}$; $X_1 = \{\langle 1,4\rangle, \langle 4,1\rangle\}$; $X_2 = \{\langle 5,1\rangle, \langle 3,2\rangle\}$; $X_3 = \{\langle 2,1\rangle\}$; $X_4 = \{\langle 1,5\rangle\}$

Proposition 4.3.1. *At least one path of T_i is strictly extended in T_{i+1}.*

Proof. Recall that $X_j \neq \emptyset$ for all j. It is clear that if all internal nodes of T_i have a child which is avoided by X_{i+1} then there is a path $a_0, \ldots, a_m$ in T_i such that X_{i+1} avoids a_j for all j.

If $T_{i+1} = T_i$ then, by construction, there is no path $a_0, \ldots, a_m$ with all of its elements avoided by X_{i+1}. Then there is an internal node of T_i, say a, with none of its children avoided by X_{i+1}. But this contradicts the badness of $\mathbf{X}$ since by construction the set of children of a is X_j for some $j \leq i$. □

As the example in Fig. 4.1 shows, the height of T_{i+1} is not necessarily greater than the height of T_i.

The following follows by construction:

Proposition 4.3.2. *Any path in T_i is a bad sequence of n-tuples with respect to the product ordering. Furthermore if $\mathbf{X}$ is g, t-controlled then any such path is $g, (t+i)$-controlled.*

Construction of the multisets M_i.

For the construction of the multisets we will use the following linearization of $(\mathbb{N}^n, \leq_{\mathrm{pr}})$ in $(\mathbb{N}^n, \leq_{\mathrm{lex}})$:

Theorem 4.3.3. *There is a function $h_n : (\mathbb{N}^n)^+ \to \mathbb{N}^n$ such that if $\mathbf{x}^\frown x$ is bad in $(\mathbb{N}^n, \leq_{\mathrm{pr}})$ and $\mathbf{x}$ is nonempty, then $h_n(\mathbf{x}^\frown x) <_{\mathrm{lex}} h_n(\mathbf{x})$. Furthermore if $\mathbf{x}$ is g, t-controlled then*

$$|h_n(\mathbf{x})|_\infty < \tilde{g}(|\mathbf{x}| - 1 + t),$$

for $\tilde{g}(x) = n!\ g(nx)^n$ and $|x|_\infty$ as in Definition 3.1.

Proof. For a proof of this theorem refer to [4]. □

Now, let $M_i \in \mathcal{M}_{<\infty}(\mathbb{N}^n)$ be defined as: $M_i(y) \stackrel{\text{def}}{=} d$ iff there are exactly d paths in T_i, say $p_1, \ldots, p_d$, such that $h_n(p_j) = y$ for all j. In other words, M_i is the multiset where we put $h_n(p)$ for every path p in T_i.

If the path $\overline{a} = a_1, \ldots, a_m$ in T_i is extended to $\overline{a}, x$ in T_{i+1} then by Theorem 4.3.3, $h_n(\overline{a}, x) <_{\mathrm{lex}} h_n(\overline{a})$. Then $M_{i+1} <_{\mathrm{ms}} M_i$. The need for working with multisets and not simply with sets resides in the fact that h is not injective.

Proposition 4.3.4. *If $\mathbf{X} = X_0, \ldots, X_k$ is g, t-controlled then $|M_k| < \tilde{g}(t+k)$, for*

$$\tilde{g}(x) = n!\ g(nx)^{n(x+1)} + 1$$

Proof. Observe that the maximum multiplicity of an element in M_k is bounded by $\prod_{j=1}^{k} g(t+j)^n \leq g(t+k)^{nk} < \tilde{g}(t+k)$. By Proposition 4.3.2 each of such path is $g, (t+k)$-controlled and by the second part of Theorem 4.3.3 we have that if $x \in M_k$ then $|x|_\infty < n!\ g(n(k+t))^n < \tilde{g}(t+k)$. □

Altogether we have shown:

Theorem 4.3.5. *There is a function $f_n : (\mathcal{P}_{<\infty}(\mathbb{N}^n))^+ \to \mathcal{M}_{<\infty}(\mathbb{N}^n)$ such that if $\mathbf{X}^\frown X$ is a bad sequence in $(\mathcal{P}_{<\infty}(\mathbb{N}^n), \leq_{\mathrm{maj}})$, $\mathbf{X}$ is nonempty and X is a nonempty set, then $f_n(\mathbf{X}^\frown X) <_{\mathrm{ms}} f_n(\mathbf{X})$. Furthermore if $\mathbf{X}$ is g, t-controlled then $|f_n(\mathbf{X})| < \tilde{g}(|\mathbf{X}| - 1 + t)$, for $\tilde{g}$ as in Proposition 4.3.4.*

Proof. Take $f_n(\mathbf{X}) = M_{|\mathbf{X}|-1}$ as in the above construction. □

Let $L^{\mathrm{maj}}_{n,g}(t)$ denote the length of the longest g, t-controlled bad sequence in $(\mathbb{N}^n, \leq_{\mathrm{maj}})$, and let $L^{\mathrm{ms}}_{n,g}(t)$ denote the length of the longest g, t-controlled decreasing sequence in $(\mathbb{N}^n, <_{\mathrm{ms}})$.

Corollary 4.3.6. *For any primitive recursive g there is a primitive recursive $\tilde{g}$ such that $L^{\mathrm{maj}}_{n,g} \leq L^{\mathrm{ms}}_{n,\tilde{g}}$. Hence there is an upper bound of $L^{\mathrm{maj}}_{n,g}$ in $\mathfrak{F}_{\omega^n}$.*

Proof. It follows from Theorem 4.3.5 and Theorem 3.3.2. □

5. APPLICATIONS

In this chapter we use our results on an upper bound for the length of the longest controlled $\leq_{\mathrm{maj}}$-bad sequence of finite subsets of $\mathbb{N}^n$ to upper bound the complexity of the emptiness problem for ITCA (§5.3) and ATRA (§5.4). Before that, in §5.1 we introduce some basic definitions regarding ITCA and in §5.2 we give the rudiments of the proof of decidability of the emptiness problem for ITCA. In these two sections we follow [21].

5.1 Incrementing tree counter automata

Without loss of generality, we will work with binary trees such that each node will either have both children or be a leaf, only non-leaf nodes will be labeled and the root node will be non-leaf.

Definition 5.1.1 (Tree)**.** A *(labeled) tree* is a tuple $\langle N, \Sigma, \Lambda \rangle$, where:

- N is a subset of $\{0,1\}^*$ such that $|N| > 1$ and N is prefix-closed, i.e. for each $n \in N$, either $n^\frown 0 \in N$ and $n^\frown 1 \in N$ or $n^\frown 0 \notin N$ and $n^\frown 1 \notin N$,
- Σ is a finite alphabet,
- Λ is a mapping from the non-leaf elements of N to Σ.

ITCA is a class of automata over trees which have natural-valued counters with increments, decrements and zero-tests. In the rest of this section we formally define ITCA.

Definition 5.1.2 (The automaton)**.** An *incrementing tree counter automaton* (ITCA) $\mathcal{C}$ with ϵ-transitions, is a tuple $\langle \Sigma, Q, q_I, F, k, \delta \rangle$ such that:

- Σ is a finite alphabet and Q is a finite set of states,
- $q_I \in Q$ is the initial state and $F \subseteq Q$ are the final states,
- $k \in \mathbb{N}$ is the number of counters,
- $\delta \subseteq (Q \times \Sigma \times L \times Q \times Q) \cup (Q \times \{\epsilon\} \times L \times Q)$ is a transition relation, where $L = \{\mathrm{inc}, \mathrm{dec}, \mathrm{ifz}\} \times \{1, \ldots, k\}$ is the instruction set.

Definition 5.1.3 (Counter valuation)**.** A *counter valuation* is just a tuple in $\mathbb{N}^k$ holding the values of the k counters at some point during execution.

For counter valuations v and v', we write:

$$v \xrightarrow{\langle \text{inc}, c\rangle}_\dagger v' \overset{\text{def}}{\Leftrightarrow} v'[c] = v[c] + 1 \wedge v'[x] = v[x] \text{ for } x \neq c$$
$$v \xrightarrow{\langle \text{dec}, c\rangle}_\dagger v' \overset{\text{def}}{\Leftrightarrow} v'[c] = v[c] - 1 \wedge v'[x] = v[x] \text{ for } x \neq c$$
$$v \xrightarrow{\langle \text{ifz}, c\rangle}_\dagger v' \overset{\text{def}}{\Leftrightarrow} v(c) = 0 \wedge v' = v$$
$$v \xrightarrow{l} v' \overset{\text{def}}{\Leftrightarrow} v \leq_{\text{pr}} v_\dagger \xrightarrow{l}_\dagger v'_\dagger \leq_{\text{pr}} v' \text{ for some } v_\dagger, v'_\dagger$$

Note that what the last definition is actually saying is that the automaton can non-deterministically increment its counters both before and after a transition. These are called *incrementing errors* and are the reason why emptiness for ITCA is decidable.

A *configuration* of $\mathcal{C}$ is a pair $\langle q, v\rangle$, where q is a state and v is a counter valuation.

A *block* is a non-empty finite sequence of configurations obtainable by performing ϵ-transitions, that is, for every two adjacent configurations $\langle q_i, v_i\rangle$ and $\langle q_{i+1}, v_{i+1}\rangle$ in a block, there exists l with $\langle q_i, \epsilon, l, q_{i+1}\rangle \in \delta$ and $v_i \xrightarrow{l} v_{i+1}$.

A *run* of $\mathcal{C}$ on a finite tree $\langle N, \Sigma, \Lambda\rangle$ is a mapping $n \mapsto B_n$ from the nodes to blocks such that:

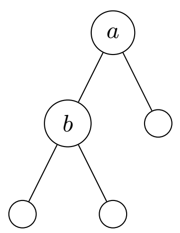
- $\langle q_I, \overline{0}\rangle$ is the first configuration in B_ϵ (here $\overline{0}$ denotes a k-tuple of 0's)
- for each non-leaf n, there exists l with $\langle q, \Lambda(n), l, r_0, r_1\rangle \in \delta$, $v \xrightarrow{l} w_0$ and $v \xrightarrow{l} w_1$, where $\langle q, v\rangle$ is the last configuration in B_n, and $\langle r_0, w_0\rangle$ and $\langle r_1, w_1\rangle$ are the first configurations in $B_{n^\frown 0}$ and $B_{n^\frown 1}$ respectively.

We regard such a run as *accepting* iff, for each leaf n, the state of the last configuration in B_n is final. The language $L(\mathcal{C})$ is the set of all finite trees with alphabet Σ on which $\mathcal{C}$ has an accepting run.

Example 5.1.4. Let $\mathcal{C} = \langle \{a, b\}, \{q_0, q_1, q_2\}, q_0, \{q_2\}, 2, \delta\rangle$ with δ being:

$$\delta \quad = \quad \{\langle q_0, a, \text{inc } 1, q_1, q_2\rangle, \langle q_1, \epsilon, \text{inc } 2, q_1\rangle \langle q_2, \epsilon, \text{dec } 2, q_2\rangle, \langle q_1, b, \text{dec } 1, q_2, q_2\rangle\}.$$

The following Figure shows an input tree T and the definitions of the blocks for certain run of $\mathcal{C}$ on input T:



$$\begin{aligned} B_\epsilon &= \langle q_0, 0, 0\rangle \\ B_0 &= \langle q_1, 1, 0\rangle, \langle q_1, 1, 1\rangle, \langle q_1, 1, 2\rangle \\ B_1 &= \langle q_2, 1, 0\rangle \\ B_{00} &= \langle q_2, 0, 2\rangle, \langle q_2, 0, 1\rangle, \langle q_2, 0, 0\rangle \\ B_{01} &= \langle q_2, 0, 2\rangle \end{aligned}$$

The leaf nodes are 1, 00 and 01. The last configuration in each B_1, B_{00} and B_{01} is in state q_2 which is final. Hence, $T \in L(\mathcal{C})$.

5.2 Emptiness for ITCA

In automata theory, the *emptiness problem* consist in determining whether the language accepted by an automaton is empty or not. In [21], this problem is proven decidable for ITCA over finite trees using the tools of the theory of well-structured transition systems [15]. In this section, we extract from such proof an explicit decision procedure, so as to make it evident how our result upper-bounds its computational complexity.

Consider an ITCA $\mathcal{C} = \langle \Sigma, Q, q_0, F, k, \delta \rangle$. For counter valuations v and v', and an instruction l, we say that v under l shields v' *lazily* iff either $v \xrightarrow{l}_{\dagger} v'$ (i.e., there are no incrementing errors) or l is of the form $\langle \text{dec}, c \rangle$, $v(c) = 0$ and $v' = v$ (i.e., decrementing a 0-valued counter leaves the array of counters unchanged).

Observe that whenever $v \leq w$ and $w \xrightarrow{l} w'$, there exists v' such that v yields v' lazily and $v' \leq w'$.

A *level* of $\mathcal{C}$ is a finite set of configurations. We denote with Levels the set of levels of $\mathcal{C}$. For levels $\mathcal{G}$ and $\mathcal{G}'$ of $\mathcal{C}$, we say that $\mathcal{G}'$ is a successor of $\mathcal{G}$ and write $\mathcal{G} \to \mathcal{G}'$ iff $\mathcal{G}'$ can be obtained from $\mathcal{G}$ as follows:

- Each $\langle q, v \rangle \in \mathcal{G}$ with $q \notin F$ is replaced either by the two configurations that some firable transition $\langle q, a, l, r_0, r_1 \rangle$ yields lazily, or by the one configuration that some firable transition $\langle q, \epsilon, l, r_0 \rangle$ yields lazily.

- Each $\langle q, v \rangle \in \mathcal{G}$ with $q \in F$ is removed.

If $K \subseteq$ Levels, we write Succ(K) for the set of immediate successors of the levels in K, i.e.

$$\text{Succ}(K) \stackrel{\text{def}}{=} \{\mathcal{G}' \in \text{Levels} \mid \exists \mathcal{G} \in K : \mathcal{G} \to \mathcal{G}'\}.$$

As usual, $\xrightarrow{*}$ denotes the transitive closure of the transition relation.

Definition 5.2.1 (A quasi-ordering between configurations $\leq$)**.** If $\langle q, v \rangle$ and $\langle r, w \rangle$ are configurations then $\leq$ is defined as

$$\langle q, v \rangle \leq \langle r, w \rangle \stackrel{\text{def}}{\Leftrightarrow} q = r \wedge v \leq_{\text{pr}} w$$

If $K \subseteq$ Levels then,

$$\uparrow K = \{\mathcal{G}' \in \text{Levels} \mid \exists \mathcal{G} \in K : \mathcal{G} \leq_{\text{maj}}^{(\leq)} \mathcal{G}'\}$$

If $K = \uparrow K$ then we say that K is upward-closed.
The following reachability algorithm decides the emptiness problem:

Algorithm 3 Decision procedure for emptiness for ITCA

initialConfiguration $\leftarrow \langle q_0, \overline{0} \rangle$
initialLevel $\leftarrow$ {initialConfiguration}
emptyLevel $\leftarrow \emptyset$
reachableLevels $\leftarrow$ {initialLevel}
newReachableLevels $\leftarrow$ {initialLevel}
repeat
    reachableLevels $\leftarrow$ newReachableLevels
    newReachableLevels $\leftarrow$ reachableLevels $\cup$ Succ(reachableLevels)
until $\uparrow$newReachableLevels $=$ $\uparrow$reachableLevels
return emptyLevel $\in$ $\uparrow$reachableLevels

5.2.1 Termination of the decision procedure

Let $X_0 \subseteq X_1 \subseteq X_2 \subseteq \ldots$. be the sequence of sets of reachable levels constructed during the execution of the algorithm, i.e.,

$$\begin{aligned} X_0 &= \{\text{initialLevel}\} \\ X_{i+1} &= X_i \cup \text{Succ}(X_i) \end{aligned}$$

The termination of the algorithm is a consequence of the following lemma:

Lemma 5.2.2 ([15, Lemma 2.4]). *The sequence $\uparrow X_0 \subseteq \uparrow X_1 \subseteq \uparrow X_2 \subseteq \ldots$ of upward-closed sets eventually stabilizes, i.e., there exists $m \in \mathbb{N}$ such that $\uparrow X_m = \uparrow X_{m+1} = \uparrow X_{m+2} \ldots$.*

Proof. Let suppose that it does not stabilize and extract an infinite subsequence where inclusion is strict: $\uparrow X_{n_0} \subset \uparrow X_{n_1} \subset \uparrow X_{n_2} \subset \ldots$. Now, for all $i > 0$ we can pick some level $\mathcal{G}_i \in X_{n_i} \setminus X_{n_{i-1}}$. Since $\leq_{\text{maj}}$ $(\leq)$ is a wqo, the infinite sequence of $\mathcal{G}_i$'s must contain an increasing pair $\mathcal{G}_i \leq_{\text{maj}}^{(\leq)} \mathcal{G}_j$ for some $i < j$. Because $\mathcal{G}_i$ belongs to an upward-closed set X_{n_i} we have that $\mathcal{G}_j$ must also be in X_{n_i} and this contradicts the assumption that $\mathcal{G}_j \notin X_{n_{j-1}}$. □

Observation 5.2.3. The m in Theorem 5.2.2 can be found effectively.

Proof. This is due to the fact that each X_i is finite and $\leq_{\text{maj}}^{(\leq)}$ is decidable. □

5.2.2 Correctness of the decision procedure

A *basis* of a upward-closed set I is a set I^b such that $I = \cup_{x \in I^b} \uparrow x$. Correctness of Algorithm 3 is grounded on the following theorems which we reproduce without proof:

Theorem 5.2.4. *The set* reachableLevels *constructed by the algorithm is a finite basis for* $\uparrow \text{Succ}^*(\{\langle q_0, \overline{0} \rangle\})$

Theorem 5.2.5. *$L(\mathcal{C})$ is non-empty iff $\emptyset \in \text{Succ}^*(\{\langle q_0, \overline{0} \rangle\})$ (i.e., the empty level is reachable from the initial level)*

Note that, since $\emptyset \leq_{\text{maj}}^{(\leq)} X \iff X = \emptyset$, then $\emptyset \in \uparrow\text{Succ}^*(\{\langle q_0, \overline{0} \rangle\}) \iff \emptyset \in \text{Succ}^*(\{\langle q_0, \overline{0} \rangle\})$.

5.3 Complexity of Emptiness for ITCA

We will extract an upper bound for the computational complexity from the proof of termination in Lemma 5.2.2.

First, note that, since $X_i = X_{i+1}$ implies $\uparrow X_i = \uparrow X_{i+1}$, and the algorithm ends when this occurs, we have that for all $i < m$, X_i is strictly included in X_{i+1}.

As in the proof of Theorem 5.2.2, let $\{\mathcal{G}_i\}$ be the sequence of levels such that for all $i > 0$, $\mathcal{G}_i \in X_i \setminus X_{i-1}$. This is a bad sequence of levels in the ordering $\leq_{\text{maj}}^{(\leq)}$. Hence, the complexity of the emptiness problem can be bounded by the length of the longest bad sequence in (Levels, $\leq_{\text{maj}}^{(\leq)}$).

As one can see, the application of Corollary 4.3.6 is not entirely straightforward because it applies to the majoring ordering of finite sets of tuples of $\mathbb{N}$ with the underlying $\leq_{\text{pr}}$ and not to levels with the underlying ordering of configurations $\leq$.

Reducing bad sequences of levels to bad sequences of finite set of tuples

We reduce bad sequences of levels to bad sequences of finite sets of tuples as follows. Suppose $Q = \{q_0, \ldots, q_{s-1}\}$ and let $q_i' \stackrel{\text{def}}{=} (i, s-i) \in \mathbb{N}^2$. Clearly if $p' \leq_{\text{pr}} q'$ then $p' = q'$ and so $p = q$. Let $\mathcal{G}$ be a level. Define

$$\mathcal{G}' \stackrel{\text{def}}{=} \{\langle p', u\rangle \in \mathbb{N}^{k+2} \mid \langle p, u\rangle \in \mathcal{G}\}.$$

The reader can verify that if $\mathcal{G}$ and $\mathcal{H}$ are levels then $\mathcal{G}' \leq_{\text{maj}}^{(\leq_{\text{pr}})} \mathcal{H}'$ implies $\mathcal{G} \leq_{\text{maj}}^{(\leq)} \mathcal{H}$. Hence $\mathbf{G} = \mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_m$, a bad sequence of levels of an ITCA with k counters, can be seen as a bad sequence of the same length $\mathbf{G}' = \mathcal{G}_0', \mathcal{G}_1', \ldots, \mathcal{G}_m'$ in $\mathcal{P}_{<\infty}(\mathbb{N}^{k+2})$ with the majoring ordering studied in Chapter 4.

A control for the bad sequence in $(\mathcal{P}_{<\infty}(\mathbb{N}^{k+2}), \leq_{\text{maj}})$

Since $\mathcal{G}_0 = \{q_0, \overline{0}\}$ then, in our construction, $\mathcal{G}_0' = \{\langle 0, |Q|-1, \overline{0}\rangle\}$. For $i > 0$, each $\mathcal{G}_i' = \{c_1, \ldots, c_{p_i}\}$ will have $k+2$-tuples where the first two components (the state part) are bounded by $|Q|$ and, at most, one of the rest of the components increments by one with respect to the generating configuration in $\mathcal{G}_{i-1}$. From Definition 3.0.1 we have that $|\mathcal{G}_i'| = \max_n\{|c_n|_\infty\}$. Hence, the bad sequence of sets is $g, 0$-controlled by $g(x) = x + 1 + |Q|$.

Now we can finally apply Corollary 4.3.6 to conclude

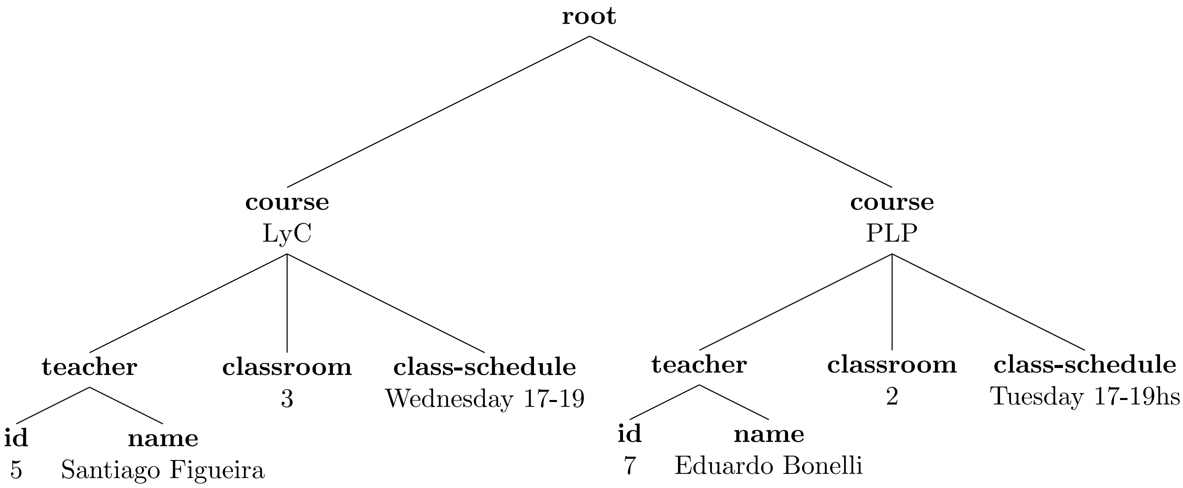
Theorem 5.3.1. *The complexity of the emptiness problem for an* ITCA *with k counters is upper bounded by a function in $\mathfrak{F}_{\omega^{k+2}}$.*

5.4 Complexity of Emptiness for ATRA

In [21], Jurdziński and Lazić also defined the Alternating Tree Register Automaton (ATRA), a top-down automaton with alternating control and one register to store and test data that runs over data trees. A data tree is simply a labeled tree whose every node carries a data value. A great variety of scenarios can be modeled with such trees and they are in close relation to XML documents. For example, consider the assignment of teachers and classrooms to courses of the computer science department. A simplified version of this problem can be modeled with trees described by the following grammar:

$$\begin{aligned} \text{root} &\to (\text{course})^* \\ \text{course} &\to (\text{teacher})^+ \;\; \text{classroom} \;\; \text{class}-\text{schedule} \\ \text{teacher} &\to \text{id} \;\; \text{name} \end{aligned}$$

An example of such trees would be:



One can specify desirable properties and restrictions of data trees —being either about the data values or the relationship between nodes— as accepting runs of ATRA.

The emptiness problem for this class of automata is proven decidable (but not primitive recursive) via a PSPACE-reduction to emptiness for ITCA (see, [21, §3]). If the ATRA $\mathcal{A}$ has s states then the ITCA $\mathcal{C}$ constructed in the reduction has $k(s) \stackrel{\text{def}}{=} 2^s - 1 + 2^{4s}$ counters. Hence, by the result of the previous section we conclude

Theorem 5.4.1. *The complexity of emptiness for an* ATRA *with s states is upper bounded by a function in $\mathfrak{F}_{\omega^{k(s)+2}}$.*

6. CONCLUSIONS

Upper bounds for controlled descending sequences in a well-order are easier to get than for controlled bad sequences in a wqo's.

We gave an upper bound for the length of controlled bad sequences over the *majoring* ordering of sets of tuples by linearizing to controlled and descending sequences of multisets with the natural *multiset* ordering. For the latter we also gave a tight upper bound, which is of interest by itself. As applications we showed complexity upper bounds for the emptiness problem for two classes of automata over trees (ITCA and ATRA).

As future work, we would like to:

- Prove, as we conjecture, that the upper bound for $L^{\mathrm{maj}}_{g,n}(t)$ is tight.
- Improve the upper bound in $\mathfrak{F}_{\omega^{k+2}}$ for ITCA. Using 2 components to represent the state seems a little too wasteful.

As another line of research, we would like to study upper bounds for the bad sequences over the dual of the majoring ordering, the minoring order:

$$A \leq^{(\leq)}_{\min} B \overset{\text{def}}{\Leftrightarrow} (\forall y \in B)(\exists x \in A)\ x \leq y.$$

This is not in general a wqo: one needs the underlying $\leq$ to be an ω^2-wqo [20, Theorem 1].

BIBLIOGRAPHY

[1] P. A. Abdulla, B. Cerans, Karlis Jonsson and Y.-K. Tsay. General decidability theorems for infinite-state systems. In *LICS*, pages 313–321, 1996.

[2] P. A. Abdulla and B. Jonsson. Ensuring completeness of symbolic verification methods for infinite-state systems. *Theor. Comput. Sci.*, 256(1-2):145–167, 2001.

[3] P. A. Abdulla and A. Nylén. Better is better than well: On efficient verification of infinite-state systems. In *LICS*, pages 132–140. IEEE Computer Society, 2000.

[4] S. Abriola, S. Figueira and G. Senno. Linearizing bad sequences: upper bounds for the product and majoring well quasi-orders. In *19th Workshop on Logic, Language, Information and Computation*, volume 7456 of *Lecture Notes in Computer Science*, pages 110–126, 2012.

[5] T. Becker and V. Weispfenning. *Gröbner Bases: A Computational Approach to Commutative Algebra*, volume 141 of *Grad. Texts in Math.* 1993.

[6] E. A. Cichoń and E. Tahhan Bittar. Ordinal recursive bounds for Higman's Theorem. *Theor. Comput. Sci.*, 201(1–2):63–84, 1998.

[7] P. Clote. On the finite containment problem for Petri nets. *Theoretical Computer Science*, 43:99–105, 1986.

[8] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theor. Comput. Sci.: Volumen B: Formal Models and Semantics*, pages 243–320. Elsevier, Amsterdam, 1990.

[9] N. Dershowitz and Z. Manna. Proving termination with multiset orderings. *Communications of the ACM*, 22(8):465–476, August 1979.

[10] D. Detlefs and R. Forgaard. A procedure for automatically proving the termination of a set of rewrite rules. In *Proc. of the first international conference on Rewriting techniques and applications*, pages 255–270, New York, NY, EEUU, 1985. Springer-Verlag New York, Inc.

[11] L. E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *The American Journal of Mathematics*, 35(4):413–422, 1952.

[12] M. R. Fellows and M. A. Langston. Nonconstructive tools for proving polynomial-time decidability. *J. ACM*, 35(3):727–739, June 1988.

[13] D. Figueira. *Reasoning on Words and Trees with Data.* Phd thesis, Laboratoire Spécification et Vérification, ENS Cachan, France, December 2010.

[14] D. Figueira, S. Figueira, S. Schmitz and P. Schnoebelen. Ackermannian and primitive-recursive bounds with dickson's lemma. In *LICS*, pages 269–278, 2011.

[15] A. Finkel and P. Schnoebelen. Well-structured transition systems everywhere! *Theoretical Computer Science*, 256(1-2):63–92, 2001.

[16] A. Finkel and P. Schnoebelen. Well-structured transition systems everywhere! *Theor. Comput. Sci.*, 256(1):2001, 1998.

[17] S. Haddad, S. Schmitz and Ph. Schnoebelen. The ordinal-recursive complexity of timed-arc Petri nets, data nets, and other enriched nets. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science (LICS'12)*, pages 355–364, Dubrovnik, Croatia, June 2012. IEEE Computer Society Press.

[18] W. Harwood, F. Moller and A. Setzer. Weak bisimulation approximants. In Z. Ésik, editor, *Computer Science Logic. 20th International Workshop, CSL 2006, 15th Annual Conference of the EACSL*, volume 4207 of *Lecture Notes in Computer Science*, pages 365–379, 2006.

[19] G. Higman. Ordering by Divisibility in Abstract Algebras. *Proc. London Math. Soc.*, s3-2(1):326–336, 1952.

[20] P. Jancar. A note on well quasi-orderings for powersets. *Information processing letters*, 72(5-6):155–160, 1999.

[21] M. Jurdziński and R. Lazić. Alternating automata on data trees and xpath satisfiability. *ACM Transactions on Computational Logic (TOCL)*, 12(3):19, 2011.

[22] J. Kruskal. *The theory of well-partially-ordered sets.* Tesis de doctorado, Princeton University, June 1954.

[23] J. Kruskal. Well-quasi-ordering, the tree theorem, and Vazsonyi's conjecture. *Trans. Amer. Math. Soc*, 95:210–225, 1960.

[24] P. Lescanne. Well rewrite orderings and well quasi-orderings. *J. Symb. Comput.*, 14(5):419–436, 1992.

[25] M. Löb and S. Wainer. Hierarchies of number theoretic functions, I. 13:39–51, 1970.

[26] K. McAloon. Petri nets and large finite sets. *Theoretical Computer Science*, 32(1–2):173–183, 1984.

[27] B. Mohar. Embedding graphs in an arbitrary surface in linear time. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, STOC '96, pages 392–397, 1996.

[28] B. Neumann. On ordered division rings. *Trans. Amer. Math. Soc.*, 66:202–346, 1949.

[29] R. Péter. *Recursive functions.* Academic Press, 1967.

[30] L. Puel. Using unavoidable set of trees to generalize Kruskal's theorem. *J. Symb. Comput.*, 8(4):335–382, 1989.

[31] R. Rado. Partial well-ordering of sets of vectors. *Mathematika*, 1:89–95, 1954.

[32] N. Robertson and P. D. Seymour. Graph minors. XX. Wagner's conjecture. *J. Comb. Theory, Ser. B*, 92(2):325–357, 2004.

[33] S. Schmitz and P. Schnoebelen. Multiply-recursive upper bounds with Higman's lemma. In *ICALP (2)*, pages 441–452, 2011.

[34] S. Schmitz. Upper bounds for Dickson's lemma. Unpublished manuscript, 2012.

[35] H. Touzet. A characterisation of multiply recursive functions with Higman's lemma. *Information and Computation*, 178(2):534–544, 2002.

[36] A. Weiermann. Complexity bounds for some finite forms of Kruskal's Theorem. *Journal of Symbolic Computation*, 18(5):463–488, 1994.