The Cantor-Bendixson Analysis of Finite Trees

Christian Wurm
cwurm@phil.uni-duesseldorf.de

Universität Düsseldorf

Abstract. We present a measure on the structural complexity of finite and infinite trees and provide some first result on its relation to context-free grammars and context-free tree grammars. In particular this measure establishes a relation between the complexity of a *language as a set*, and the complexity of the *objects it contains*. We show its precise nature and prove its decidability for the formalisms we consider.

1 Introduction

We introduce a measure on the complexity of trees, working equally well on finite and infinite trees. This finitary Cantor-Bendixson rank is a slight adaptation from the concept of Cantor-Bendixson rank (CB-rank), which is however only meaningful on infinite trees. Our modification makes the concepts relevant to the theory of formal languages and grammars, a field where it seems to be unknown so far.

The finitary CB-rank (henceforth: FCB-rank) provides a measure of the complexity of trees, as they are for example used as representatives of derivations of sentences/words in formal language theory. Its main advantage is: it is based on discrete structural properties and thus highly informative on the structure of a tree; and at the same time, it is insensitive to things as unary branches, binary versus ternary branching etc. *Prima facie*, the FCB-rank does not say anything about the complexity of a certain grammar, only about certain objects it generates. We can refer to the latter as *syntagmatic complexity*, as a property of a single object of a language, to the former as *paradigmatic complexity*, as a property of the set as an entire collection. There is usually no immediate relation between the two – simple languages as Σ^* can be generated by complicated grammars.[1] Using the FCB-rank, we can establish this sort of relation: we will assign an FCB-rank to context-free grammars, and show that this rank corresponds exactly to the hierarchy of k-linear languages. Moreover, we show that the rank of a grammar can be effectively computed.

As we said, we relate the theory of formal grammars/languages to the theory of relations/structures. This means we can talk about trees regardless of their mode of generation. So it does not matter whether we talk about generation trees of context-free grammars or trees generated by regular tree grammars, tree

[1] With the well-known consequence that the universality problem is undecidable for CFG.

adjoining grammars, context-free tree grammars etc. We show how the FCB-rank can be computed for (simple) context-free tree grammars (CFTG). Whereas for CFG, bounded FCB-rank coincides with an established class of grammars, for CFTG this allows us to define new classes of tree- and string languages.

2 Definitions: Trees, Derivatives, Cantor-Bendixson rank

We will present trees as structures in a well-known fashion; we take the concepts of linear order, partial order for granted.[2]

Definition 1 *A well-founded tree is a structure $(T, \trianglelefteq)$, where T is a set (the set of nodes) and $\trianglelefteq \subseteq T \times T$ is a partial order with a smallest element r and with the property that for each $t \in T$, the set $\{s : s \trianglelefteq t\}$ is 1. finite and 2. linearly ordered by $\trianglelefteq$.*

If $s \trianglelefteq t$, we say s dominates t. We denote the non-reflexive restriction of $\trianglelefteq$ by $\triangleleft$, and the immediate dominance relation by $\triangleleft_i$, defined by $x \triangleleft_i y :\Longleftrightarrow x \triangleleft y$ and $x \trianglelefteq z \trianglelefteq y \to x = z$ or $y = z$. Note that a tree in this sense does not yet have a precedence order on the nodes which are not ordered by dominance; what we have specified is only the dominance order. We will throughout this paper assume that trees are well-founded.

A **path** in a tree $\mathcal{T}$ (or $\mathcal{T}$-path) is a set $P \subseteq T$ which is linearly ordered by $\trianglelefteq$ and convex.[3] A **complete $\mathcal{T}$ path** is a path which is maximal wrt. inclusion, that is, is not a proper subset of any other $\mathcal{T}$-path. The **depth** of a tree $depth(\mathcal{T})$ is the length of its longest path, where by length of a path P - as paths are sets - we mean its cardinality $|P|$. If P is a complete path in $\mathcal{T}$ with $t \in P$, t is $\trianglelefteq$-maximal in P, then we also say that $|P|$ is the depth of t ($depth(t)$), with $\mathcal{T}$ intended. By a **chain** we denote a tree $(T, \trianglelefteq)$ where T is linearly ordered by $\trianglelefteq$. A **full n-ary tree** is a tree in which every node has zero or n children. A **complete n-ary tree** is a full n-ary tree, such that if there is a leaf t with $depth(t) = k$, then for all $t' \in T$, $depth(t') = k$ if and only if t' is a leaf (this covers the case where there are no leaves).

Definition 2 *Let $\mathcal{T} = (T, \trianglelefteq)$ be a tree. We define the **restriction** $\mathcal{T} \restriction S$ of $\mathcal{T}$ to $S \subseteq T$ as $(S, \trianglelefteq_{\restriction S})$, where $\trianglelefteq_{\restriction S} := \trianglelefteq \cap (S \times S)$.*

We say that $\mathcal{S} = (S, \trianglelefteq)$ is a **subtree** of $\mathcal{T} = (T, \trianglelefteq)$, if $\mathcal{S}$ is a tree, and $\mathcal{S} = \mathcal{T} \restriction S$ for some $S \subseteq T$. We slightly abuse notation and write $\mathcal{S} \subseteq \mathcal{T}$ for the subtree relation; similarly, if $t \in T$, we also write $t \in \mathcal{T}$. [4]

[2] For a good general introduction into the theory of relations and structures, consider [3]

[3] A subset P of a partially ordered set $(Q, \leq)$ is convex, if from $x, y \in P$, $x \leq z \leq y$ it follows that $z \in P$.

[4] Note that, if we think of trees as graphs, a subtree of $\mathcal{T}$ need not be a contiguous subgraph of the graph of $\mathcal{T}$. Our definition of subtree is, up to isomorphism, equivalent to an order theoretic definition, which says: $\mathcal{S} \subseteq \mathcal{T}$ if there is an order embedding $i : \mathcal{S} \to \mathcal{T}$. Then $\mathcal{S}$ is the isomorphic copy of a subtree of $\mathcal{T}$.

Definition 3 *The* ***core*** $c(\mathcal{T})$ *of a tree* $\mathcal{T}$ *is the set of nodes which lie on two distinct complete paths of the tree:* $c(\mathcal{T}) := \{t \in T$ *: there are complete* $\mathcal{T}$*-paths* $P_1, P_2 \subseteq T$*,* $t \in P_1 \cap P_2$*, and* $P_1 \neq P_2\}$*. Define the* ***derivative*** *of a tree as* $d(\mathcal{T}) = \mathcal{T} \restriction c(\mathcal{T})$*. Furthermore, for* $n \in \mathbb{N}$*,* $d^n(\mathcal{T}) = d(d^{n-1}(\mathcal{T}))$*, where* $d^0(\mathcal{T}) = \mathcal{T}$*.*

The derivative of a tree cuts away all nodes which lie only on a single complete path. $\mathcal{T}_\emptyset$ denotes the empty tree, i.e. the tree with empty domain.

Definition 4 *For* $\mathcal{T}$ *a tree, the finitary CB-rank* $FCB(\mathcal{T})$ *is defined as the least natural number* n *such that* $d^n(\mathcal{T}) = \mathcal{T}_\emptyset$*. We put* $FCB(\mathcal{T}) = \omega$*, if there is no* $n \in \mathbb{N}$ *such that* $d^n(\mathcal{T}) = \mathcal{T}_\emptyset$*.*

Some observations: every finite tree has a finite FCB-rank, and every finite tree will eventually converge to $\mathcal{T}_\emptyset$. If $\mathcal{S} \subseteq \mathcal{T}$, then $d(\mathcal{S}) \subseteq d(\mathcal{T})$, and so $FCB(\mathcal{S}) \leq FCB(\mathcal{T})$. Furthermore, if $s, t \in \mathcal{T}$, $s \trianglelefteq t$ and $t \in d(\mathcal{T})$, then $s \in d(\mathcal{T})$. There are infinite trees $\mathcal{T}$ for which $FCB(\mathcal{T})$ is finite. $FCB(\mathcal{T})$ can be infinite in two distinct cases: either there is an $n \in \mathbb{N}$ such that $d^n(\mathcal{T}) = d^{n+1}(\mathcal{T}) \neq \mathcal{T}_\emptyset$; that is, our derivatives reach a fixed point. The other possibility is that for all $n \in \mathbb{N}$, $d^n(\mathcal{T}) \supsetneq d^{n+1}(\mathcal{T})$, which entails that for all $n \in \mathbb{N}$, $d^n(\mathcal{T}) \neq \mathcal{T}_\emptyset$. If $FCB(\mathcal{T}) = n$, then for any $m \leq n$, there is $\mathcal{S} \subseteq \mathcal{T}$ with $FCB(\mathcal{S}) = m$.

The Cantor-Bendixson rank is originally used with infinite trees, where the *core* is defined as in definition 3 with the additional requirement that paths be infinite (see for example [10],[12]); and $CB(\mathcal{T})$ is the least ordinal α such that $d^\alpha(\mathcal{T}) = d^{\alpha+1}(\mathcal{T})$. Skipping the infinity condition makes this notion applicable in the finite; the other modification allows us to avoid some unwanted conclusions in the infinite.[5] Note that however our concept works equally well with infinite trees. Consider some examples:

1. For the countably infinite complete binary tree $\mathcal{T}$, we have $FCB(\mathcal{T}) = \omega$, because $d(\mathcal{T}) = \mathcal{T}$.
2. Let $\mathcal{T}$ be the finite or infinite derivation tree of a regular string grammar. Then regardless of $depth(\mathcal{T})$, $FCB(\mathcal{T}) = 2$.
3. Let $\mathcal{T}$ be the complete, at least binary branching tree of depth k. Then $FCB(\mathcal{T}) = k$.
4. Let $\mathcal{T}$ be a tree of the form $\mathcal{T}_1[\mathcal{T}_2[\mathcal{T}_3[...]]]$, where $\mathcal{T}_i$ is the complete binary tree of depth i, and the root of $\mathcal{T}_{i+1}$ is immediately dominated by an arbitrary leaf of $\mathcal{T}_i$. Then we have for all $n \in \mathbb{N}$, $d^n(\mathcal{T}) \neq d^{n+1}(\mathcal{T})$, and so we have $FCB(\mathcal{T}) = \omega$.
5. If $FCB(\mathcal{T})$ is finite, then $d^{FCB(\mathcal{T})}(\mathcal{T}) = \mathcal{T}_\emptyset$.

The following lemma will help in understanding the concept, and in some sense generalizes the above examples:

[5] Under the definition of CB-rank, the countable infinite complete binary has CB-rank 0, contrary to finite trees. So in the infinite, we lose the property of monotonicity: $\mathcal{S} \subseteq \mathcal{T} \not\Rightarrow FCB(\mathcal{S}) \leq FCB(\mathcal{T})$.

Lemma 5 *For $k \in \mathbb{N}$, $FCB(\mathcal{T}) \geq k$, if and only if there is a complete binary branching subtree of $\mathcal{T}$ with depth k.*

Proof. *If*: Easy exercise.

Only if: by induction on the (inverse) derivation. Assume that $FCB(\mathcal{T}) \geq k$. We have $\mathcal{T}_\emptyset \subseteq d^k(\mathcal{T}) \subsetneq d^{k-1}(\mathcal{T})$. So $d^{k-1}(\mathcal{T})$ must contain a complete binary tree of depth ≥ 1 (complete binary is trivially satisfied for depth 1, that is, the singleton tree). Next we make the induction step: assume that for $1 \leq i < k$, $d^{k-i}(\mathcal{T})$ contains a full binary tree of depth i. As $\trianglelefteq$ is acyclic and we are in the finite, we have a set L of leaves, which are maximal with respect to $\trianglelefteq$ in $d^{k-i}(\mathcal{T})$. As $d^{k-i}(\mathcal{T}) \neq d^{k-(i+1)}(\mathcal{T})$, these nodes must have been in the core of $d^{k-(i+1)}(\mathcal{T})$. Consequently, for each $l \in L$, there must have been two nodes m, n, such that $l \trianglelefteq m$, $l \trianglelefteq n$, and m, n are on distinct paths. Given these nodes, we know that $d^{k-(i+1)}(\mathcal{T})$ contains the full binary tree of depth $i + 1$. □

Another observation we now can make is the following: if $d(\mathcal{T}) = \mathcal{T}$, then either $\mathcal{T} = \mathcal{T}_\emptyset$, or $\mathcal{T}$ is infinite, more precisely: every node in $\mathcal{T}$ dominates a complete infinite binary subtree.

We will not ponder on questions of psycholinguistic nature, but for completeness of exposition, we will very shortly illustrate what in our view are the major advantages of the FCB-rank with respect to other current (linguistic) methods of measuring the complexity of (finite) trees (see for example [5]). (1) FCB-rank has values in $\mathbb{N} \cup \{\omega\}$. (2) FCB-rank is computed locally from the most complex subtree (recall Lemma 5). (3) Given the FCB-rank of a tree $\mathcal{T}$, we can precisely say what kind of subtrees do *not* occur in $\mathcal{T}$, and which ones *do*, and which ones *can*; the measure is very informative about structure. (4) The FCB-rank is monotonous over the subtree relation: if $\mathcal{S} \subseteq \mathcal{T}$, then $FCB(\mathcal{S}) \leq FCB(\mathcal{T})$. This does not hold for many other measures, as can be easily shown. (5) FCB rank is insensitive to unary branching; moreover a tree of the form [[..][..][..]] [6] and a tree of the form [[..][[..][..]]] get the same FCB rank. In our view, this point speaks in favor of FCB-rank, as these are so to speak variants encoding the same dependencies.

3 Ordered Trees and Labelled Trees

The trees we presented in the previous section were unordered; there was no precedence specified between elements not dominating each other. Trees generated by phrase structure grammars usually have a precedence order specified as an intrinsic feature.

Definition 6 *A well-founded ordered tree is a structure $(T, \trianglelefteq, \preceq)$, where $(T, \trianglelefteq)$ is a well-founded tree, $\preceq \subseteq T \times T$ is a partial order, for each $t \in T$, $t \trianglelefteq t'$, $\{s : t \lhd_i s\} \cap \{s : s \preceq t'\}$ is finite, and where*

[6] For example linguists sometimes dislike such subtrees, for rather ideological reasons as X-Bar theory or assumption of binary branching

1. *for arbitrary elements s, t, we have $s \preceq t$ or $t \preceq s$ if and only if $s \not\trianglelefteq t$ and $t \not\trianglelefteq s$;*
2. *if $s \preceq t$, then for all u, v, if $s \trianglelefteq u, t \trianglelefteq v$, then $u \preceq v$.*

We write an ordered tree as $(\mathcal{T}, \preceq)$, but in the sequel also simply as $\mathcal{T}$, as long as misunderstandings can be excluded. The reason is that our definition of a FCB-rank can be equally well applied to ordered trees as to trees. Define $d(\mathcal{T}, \preceq) = (d(\mathcal{T}), \preceq_{\restriction d(\mathcal{T})})$. We can easily check that the derivative of an ordered tree $(\mathcal{T}, \preceq)$ is an ordered tree, and $FCB(\mathcal{T}, \preceq) = FCB(\mathcal{T})$. So the order $\preceq$, to which we also refer as precedence, does not play any role for the derivative.

An (ordered) **labelled tree** is a structure $(\mathcal{T}, X_1, ..., X_n)$, where $\mathcal{T}$ is an (ordered) tree, and $X_1, ..., X_n \subseteq T$, such that for $i \neq j$, $X_i \cap X_j = \emptyset$, and $\bigcup_{1 \leq i \leq n} X_i = T$.[7] $X_1, ..., X_n$ are the labels of $\mathcal{T}$. Given a labelled tree $\mathcal{T} = (T, \trianglelefteq, (\preceq), X_1, ..., X_n)$, $U \subseteq T$, we define $\mathcal{T} \restriction U := (U, \trianglelefteq_{\restriction U}, (\preceq_{\restriction U}), X_1 \cap U, ..., X_n \cap U)$. This is again an (ordered) labelled tree, so labels have no influence on core, derivatives etc., and so we can define the FCB-rank of labelled trees in the usual fashion.

4 Context-free Grammars and Derivation Trees

Context-free grammars are very well-known; so we just fix our conventions. A context-free grammar (CFG) is a tuple $(\mathtt{S}, \mathcal{N}, A, R)$ with $\mathtt{S} \in \mathcal{N}$ (note the font to avoid confusion), $\mathcal{N}$ the non-terminals, A the set of terminals, and $R \subseteq \mathcal{N} \times (\mathcal{N} \cup A)^*$ the set of production rules. To stick with conventional notation, we write $N \to \alpha$, if $(N, \alpha) \in R$; we assume lowercase Greek letters to be ranging over $(\mathcal{N} \cup A)^*$. The sets $A, \mathcal{N}, R$ are supposed to be finite. Derivability is defined as follows: let $\vdash_G \subseteq (\mathcal{N} \cup A)^* \times (\mathcal{N} \cup A)^*$ be defined by: if $N \to \alpha \in R$, then for all $\beta_1, \beta_2 \in (\mathcal{N} \cup A)^*$, $(\beta_1 N \beta_2, \beta_1 \alpha \beta_2) \in \vdash_G$, and nothing else. Let $\vdash_G^*$ be the reflexive and transitive closure of $\vdash_G$. We stick to common usage and write $\alpha \vdash_G^* \beta$ instead of $(\alpha, \beta) \in \vdash_G^*$.

We now define how what is usually called the "derivation tree" of a CFG relates to the relational notion of trees. Given a tree $\mathcal{T}$, its **elementary subtrees** are its subtrees $\mathcal{S}$ of depth 2, where for t the root of $\mathcal{S}$, $t' \neq t$, $t' \in S$ if and only if $t \triangleleft_i t'$ in $\mathcal{T}$. Given a CFG $G = (\mathtt{S}, \mathcal{N}, A, R)$, we say a well-founded, ordered labelled tree $\mathcal{T}_L = (T, \trianglelefteq, \preceq, (N)_{N \in \mathcal{N}}, (a)_{a \in A})$ is a **derivation tree** of G, if for r the unique $\trianglelefteq$-minimal element of the tree, we have $r \in \mathtt{S}$ (henceforth, we will stick to the convention that r denotes the root), and for each elementary subtree $\mathcal{T} \subseteq \mathcal{T}_L$, $T = \{s_1, ..., s_i\}$, $s_1 \trianglelefteq s_2, ..., s_i$ and $s_2 \prec ... \prec s_i$, we have $s_1 \in X_1, ..., s_i \in X_i$ only if there is a rule $X_1 \to X_2 ... X_i \in R$. We denote the set of derivation trees of a grammar G by $D(G)$. If we omit the condition that $r \in \mathtt{S}$, we say the tree is G-**conform** (note that we omit the condition that leaves have terminal labels; this will have some importance in the sequel). What is crucial

[7] In words: every $t \in T$ has exactly one label. One sometimes assumes (equivalently) a labelling function; we choose another way to stay within the boundaries of strict relation theory.

for connecting results from formal language theory to structure theory is the connecting lemma, for which we omit the proof.

Lemma 7 *(Connecting Lemma) Let $G = (S, \mathcal{N}, A, R)$ be a CFG, $X_0, ..., X_i \in \mathcal{N}$. We have $\mathcal{T} \in D(G)$ if and only if for all subtrees $\mathcal{S} \subseteq \mathcal{T}$ the following holds: for all $s, t_1, ..., t_i \in S$, where s is the root of $\mathcal{S}$ and $t_1 \prec ... \prec t_i$, we have $s \in X_0, t_1 \in X_1, ..., t_i \in X_i$ if and only if $X_0 \vdash_G^* \alpha_1 X_1 \alpha_2 ... \alpha_i X_i \alpha_{i+1}$.*

The proof is by induction on the number of derivation steps. This notion thus captures our intuition on derivation trees, except for one important detail: we also allow for infinite derivation trees. This will be important for what is to follow, because we are interested in grammars generating trees with unbounded FCB-rank.[8]

5 Decidability Results

We now define what it means for a CFG to have a certain FCB-rank; given a set M of ordinals, by the supremum M, in symbols $sup(M)$, we denote the smallest ordinal which is larger than (or equal to) all elements of M.

Definition 8 *Given a CFG G, define the FCB rank of G by $FCB(G) := sup\{FCB(\mathcal{T}) : \mathcal{T} \in D(G)\}$.*

In our case, the supremum actually coincides with the maximum: if a CFG has derivation trees of arbitrarily large FCB-rank, it has a derivation tree of infinite rank (we prove this later on, lemma 11). Given a grammar G, its FCB-rank can be finite or infinite, but it is well-defined in any case. The first main result shows that we can effectively compute the FCB-rank of a CFG. But first we need to introduce a construction which is essential for the proof of the theorem.

Let I be an arbitrary index set, $\{\mathcal{T}_i : i \in I\}$ be a set of pairwise disjoint trees.[9] By $\otimes\{\mathcal{T}_i : i \in I\}$ we denote the tree $(\bigcup_{i \in I} T_i \cup \{r\}, \trianglelefteq)$, where $r \notin \bigcup_{i \in I} T_i$ and for $s \neq r \neq t$, $s \trianglelefteq t$ if and only if $s \trianglelefteq_{T_i} t$ for some $i \in I$, and $r \trianglelefteq s$ for all $s \in (\bigcup_{i \in I} T_i \cup \{r\})$. By convention, we put $\otimes\emptyset = \otimes\{\mathcal{T}_\emptyset\} = (\{r\}, \{\langle r, r \rangle\})$, the singleton tree. The proof of the first main theorem relies on the following simple, beautiful property, which also illustrates the "structural locality" of the notion of core and derivative.

Lemma 9 *(Locality Lemma) Assume $\{\mathcal{T}_i : i \in I\}$ contains at least 2 non-empty trees. Then $d(\otimes\{\mathcal{T}_i : i \in I\}) = \otimes\{d[\{\mathcal{T}_i : i \in I\}]\} := \otimes\{\{d(\mathcal{T}_i) : i \in I\}\}$*

[8] For the reader who distrusts in infinite trees (as does the author) I should add: we could also restrict ourselves to finite trees, and then the statement: "there is a $\mathcal{T} \in D(G)$ with infinite FCB-rank" translates: "there is no finite upper bound to the FCB-rank of finite trees $\mathcal{T} \in D(G)$", and the two statements can be shown to be equivalent. But the latter usually requires an additional step, so using infinite trees is a matter of convenience.

[9] We use this as a shorthand for: trees with disjoint domains. In the sequel, for this and similar constructions we will always assume that trees are disjoint.

Proof. 1. $d(\otimes\{\mathcal{T}_i : i \in I\}) \subseteq \otimes\{d[\{\mathcal{T}_i : i \in I\}]\}$. Assume $t \in d(\otimes\{\mathcal{T}_i : i \in I\})$. Assume $t \in \mathcal{T}_i$ for some $i \in I$. Then there are two distinct paths P_1, P_2 starting in t and each containing a leaf, where all elements in this paths are in $\mathcal{T}_i$; therefore, $t \in d(\mathcal{T}_i)$, and so $t \in \otimes\{d[\{\mathcal{T}_i : i \in I\}]\}$. Otherwise, assume we have $t = r$. Then by definition, $r \in \otimes\{d[\{\mathcal{T}_i : i \in I\}]\}$.

2. $\otimes\{d[\{\mathcal{T}_i : i \in I\}]\} \subseteq d(\otimes\{\mathcal{T}_i : i \in I\})$. Assume $t \in \otimes\{d[\{\mathcal{T}_i : i \in I\}]\}$. Then either $t \in c(\mathcal{T}_i)$ for some $i \in I$, and so $t \in c(\otimes\{\mathcal{T}_i : i \in I\})$, because the core is monotonous over the subtree relation. Or $t = r$. As $\{\mathcal{T}_i : i \in I\}$ contains two non-empty trees, there are at least two complete paths in $\otimes\{\mathcal{T}_i : i \in I\}$, and so $r \in d(\otimes\{\mathcal{T}_i : i \in I\})$.

This shows equality of the domains; we skip the proof of equality of relations, which follows in a straightforward fashion.[10] □

Corollary 10 *Let $\{\mathcal{T}_i : i \in I\}$ be a set of trees. Put $k = max\{FCB(\mathcal{T}_i) : i \in I\}$.*

1. *Assume there are $\mathcal{T}_i, \mathcal{T}_j : i \neq j, i, j \in I$, such that $FCB(\mathcal{T}_i) = FCB(\mathcal{T}_j) = k$. Then $FCB(\otimes\{\mathcal{T}_i : i \in I\}) = k + 1$.*
2. *Assume there are no $\mathcal{T}_i, \mathcal{T}_j : i \neq j, i, j \in I$, such that $FCB(\mathcal{T}_i) = FCB(\mathcal{T}_j) = k$. Then $FCB(\otimes\{\mathcal{T}_i : i \in I\}) = k$.*

Proof. 1. Assume we have $FCB(\mathcal{T}_i) = FCB(\mathcal{T}_j) = k$. Then $d^{k-1}(\mathcal{T}_i), d^{k-1}(\mathcal{T}_j)$ are two non-empty chains. As they are disjoint, there are two complete paths through r in $d^{k-1}(\otimes\{\mathcal{T}_i : i \in I\}) = \otimes(d^{k-1}[\{\mathcal{T}_i : i \in I\}]$. Therefore, $d^k(\otimes\{\mathcal{T}_i : i \in I\}) = (\{r\}, \trianglelefteq_{\restriction\{r\}})$, and so, $FCB(\otimes\{\mathcal{T}_i : i \in I\}) = k + 1$.

2. Assume there is only one $\mathcal{T} \in \{\mathcal{T}_i : i \in I\}$ such that $FCB(\mathcal{T}) = k$, and all other trees have strictly smaller FCB-rank. It follows that for $s \notin \mathcal{T} \cup \{r\}$, we have $s \notin \otimes(d^{k-1}[\{\mathcal{T}_i : i \in I\}]) = d^{k-1}(\otimes\{\mathcal{T}_i : i \in I\})$. Moreover, $d^{k-1}(\mathcal{T})$ is a chain; so $d^{k-1}(\otimes\{\mathcal{T}_i : i \in I\}) = \otimes\{d^{k-1}(\mathcal{T})\}$, which is also a chain; so $d^k(\otimes\{\mathcal{T}_i : i \in I\}) = \mathcal{T}_\emptyset$. □

Lemma 11 *(1) If for a CFG $G = (\mathtt{S}, \mathcal{N}, A, R)$, we have $FCB(G) = \omega$, then there is $\mathcal{T} \in D(G)$ such that the complete infinite binary tree is a subtree of $\mathcal{T}$; moreover, (2) there is a reachable $N \in \mathcal{N}$ such that $N \vdash^*_G \alpha N \beta N \gamma$ for some $\alpha, \beta, \gamma \in (\mathcal{N} \cup A)^*$.*

Proof. We first prove (2) by contradiction. Assume $FCB(G) = \omega$. Then for any $k \in \mathbb{N}$, there is $\mathcal{T} \in D(G)$ with $FCB(\mathcal{T}) \geq k$. Choose $\mathcal{T}$ with $FCB(\mathcal{T}) \geq |\mathcal{N}| + 1$. Then take the root r with label $\mathtt{S}$. r dominates two subtrees $\mathcal{S}_1, \mathcal{S}_1'$ of rank $\geq |\mathcal{N}|$ (see lemma 13). If nodes with label $\mathtt{S}$ occur in both $\mathcal{S}_1, \mathcal{S}_2$, then (2) follows; if not, there is a subtree of rank $\geq |\mathcal{N}|$, in which only $|\mathcal{N}| - 1$ labels occur ($\mathtt{S}$ does not). Take its root t with label X. It dominates two subtrees $\mathcal{S}_2, \mathcal{S}_2'$ of rank $\geq |\mathcal{N}| - 1$. If X occurs in both $\mathcal{S}_2, \mathcal{S}_2'$, then (2) follows; if not, there is a subtree of rank $\geq |\mathcal{N}| - 1$, in which only $|\mathcal{N}| - 2$ labels occur ($\mathtt{S}, X$ do not). We iterate this, until we are left with a subtree of rank ≥ 1, in which 0 labels occur

[10] Note that if $\{\mathcal{T}_i : i \in I\}$ does not contain two non-empty trees, the result does not obtain: let $\mathcal{C}$ be a chain. $d(\otimes\{\mathcal{C}\}) = \mathcal{T}_\emptyset$, and $\otimes(d(\mathcal{C})) \neq \mathcal{T}_\emptyset$.

- contradiction, as our labelling is exhaustive. Thus there are $t, t', t'' \in \mathcal{T}$ with $t, t', t'' \in X$ for some label X, $t \trianglelefteq t', t''$ and $t' \prec t''$. By the connecting lemma, this holds if and only if $X \vdash_G^* \alpha X \beta X \gamma$.

Now (1) follows easily: by $\mathtt{S} \vdash_G^* \alpha N \beta$, $N \vdash_G^* \alpha N \beta N \gamma$ and the connecting lemma we construct a tree having the complete infinite binary tree as a subtree. Take $\mathcal{T}$, in which there are infinitely many distinct $t, t', t'' \in \mathcal{T}$ such that $t, t', t'' \in N$, and $t \lhd t', t''$ and $t' \prec t''$ (this exists by the connecting lemma). Then there is a subtree $\mathcal{S} \subseteq \mathcal{T}$ which consists only of the nodes $s \in N$; this is a complete infinite binary tree. □

Theorem 12 *Given a context-free grammar G, there is an algorithm which computes $FCB(G)$.*

Proof. Part 1: the algorithm

Assume without loss of generality that every non-terminal is reachable; all those who are not can be thrown out anyway.

a) The infinite case: we check whether $FCB(G) = \omega$.

It is well-known that we can compute for any $N \in \mathcal{N}$ the set $der_G^1(N) := \{M : N \vdash_G^* \alpha M \beta\}$.[11] From this it follows that we can also compute $der_G^2(N) := \{(N_1, N_2) : N \vdash_G^* \alpha N_1 \beta N_2 \gamma\}$; we quickly explain the proof: we transform a grammar G to G' by adding non-terminals in $\mathcal{N} \times \mathcal{N}$, and rules $N \to (N_1, N_2)$ if $N \vdash_G \alpha N_1 \beta N_2 \gamma$ (NB: $\vdash_G$ is not the reflexive and transitive!); furthermore, if $N_1 \vdash_G^* \alpha N_3 \beta$, $N_2 \vdash_G^* \alpha' N_4 \beta'$, then we add rules $(N_1, N_2) \to \{(X, Y) : X \in \{N_1, N_3\}, Y \in \{N_2, N_4\}\}$. This procedure terminates, because the set of possible pairs is finite, and we can compute $der_G^1(N)$ for each $N \in \mathcal{N}$; the only point where we introduce new pairs is for the immediate derivability $\vdash_G$, which is trivially decidable. Call the resulting grammar G'. It can be shown that $(N_1, N_2) \in der_G^2(N)$ if and only if $(N_1, N_2) \in der_{G'}^1(N)$. Now if in a grammar $G = (\mathtt{S}, \mathcal{N}, A, R)$ we have an $N \in \mathcal{N}$, such that $(N, N) \in der_G^2(N)$, then we put $FCB(G) = \omega$.

b) The finite case. If $FCB(G) \neq \omega$, proceed as follows:

1. Pick out all G-rules of the form $N \to \alpha a \beta$ where $a \in A$. If $\alpha\beta \neq \epsilon$, put $val(N) := 2$, otherwise, $val(N) := 1$.

2. If there is a non-terminal N such that $N \to M \in R$ for some M such that $val(M) := n$, $val(N) \leq n$, we put $val(N) := n$.

3. If there is an N such that $N \to \alpha M \beta O \gamma \in R$, where $min\{val(M), val(O)\} \geq n$, and $val(N) \leq n$, we put $val(N) := n + 1$.

4. Iterate steps 2 and 3 until we cannot assign any higher value to any non-terminal.

5. Put $FCB(G) = val(\mathtt{S})$.

Importantly, this procedure will converge only if there is no $N \in \mathcal{N}$ such that $N \vdash_G^* \alpha N \beta N \gamma$; so we first have to check whether $FCB(G) = \omega$.

Part 2: Correctness and completeness

[11] For a more recent reference on decision problems of CFG, consider [8].

a) The infinite case

Completeness: It follows from lemma 11 (2) that if $FCB(G) = \omega$, we have $N \in \mathcal{N}$ with $N \vdash_G^* \alpha N \beta N \gamma$, which is our algorithm can decide.

Correctness: If for $G = (\mathtt{S}, \mathcal{N}, A, R)$, $N \in \mathcal{N}$, we have $N \vdash_G^* \alpha N \beta N \gamma$, such that our algorithm assigns the grammar the FCB-rank ω, then there is $\mathcal{T} \in D(G)$ with $FCB(\mathcal{T}) = \omega$; see the construction in proof of lemma 11 (1).

b) The finite case

Assume $FCB(G) \neq \omega$. Then we can establish by induction that we get the correct rank: we assign the correct rank to the elementary trees which are conform to a grammar G (that is, 1 or 2). And if we assign the correct values to the elementary trees, then we assign the correct values to larger G-conform trees when we construct G-conform trees with the $\otimes$-method, as follows from corollary 10. Moreover, as the number of non-terminals is finite, the possible trees of relevant (binary) structure become less and less, otherwise we have $FCB(G) = \omega$; so the procedure terminates. □

6 Expressive Power

The next question is: what is the class of languages for which there is a CFG with bounded FCB-rank?[12] Recall that a context-free grammar is **linear**, if every rule in R has the form: $N \to \alpha N' \beta$ with $\alpha, \beta \in A^*$. A language is linear if it is generated by some linear CFG. Obviously, linear grammars are related to bounded FCB-rank: L is linear if and only if there is a grammar G such that $L(G) = L$ and $FCB(G) \leq 2$. This is fairly straightforward. But note that there are grammars G with $FCB(G) \leq 2$ which are not linear: we might have a rule $\mathtt{S} \to N_1 N_2 N_3$, where N_1, N_3 only generate unary branches. These can of course be brought into linear form; but we have to be careful in distinguishing properties of grammars and languages. For simplicity, we will mostly assume that our CFGs do not have unary rules; call such grammars *neat.* Then we can easily check that: G is linear if and only if G is neat and $FCB(G) \leq 2$. We now generalize this result. Let A be an alphabet. A **(linear) substitution** is a map $\sigma : A \to \{L_a : a \in A\}$, where each L_a is a (linear) language; this map is extended to strings and sets in the usual fashion, and can be applied to arbitrary languages $L \subseteq A^*$ (we use square brackets $f[-]$ to indicate the pointwise extension of a function to subsets of the domain. In general, if the $L_a : a \in A$ are all contained in a class of languages $\mathcal{C}$, then we say σ is a substitution into $\mathcal{C}$. A **linear grammar substitution** $G = (\mathtt{S}, \mathcal{N}, A, R)$ by a set $\{G_a : a \in A\}$ of linear grammars is obtained by substituting all occurrences of a terminal a in R by $\mathtt{S}_{G_a}$, and then taking the union of the rules of $\{G_a : a \in A\}$ with the modified rules of G. Obviously, there is a close correspondence of linear substitutions and linear grammar substitutions. As this is obvious and well-known, we will

[12] In this section, we presuppose familiarity of the reader with standard techniques in formal language theory. For those who want some background in this topic, we refer to [1],[8].

be sometimes a bit sloppy when moving from grammars to languages and vice versa, avoiding some tedious details.

Call a language of the form $\sigma_{k-1}[...\sigma_1[L]]$, where L is linear and $\sigma_1, ..., \sigma_{k-1}$ are linear substitutions, k-**linear**; we denote the class of k-linear languages by k-**LIN**. The hierarchy of k-linear languages is well-known (see [1],p.209), and it has been shown in [4] that these classes form a proper infinite hierarchy.[13] Before we can present the main theorem connecting the FCB-rank with k-**LIN**, we need to establish some additional results.

Given a tree $\mathcal{T}$, $t \in \mathcal{T}$ a node, we define $rank(t, \mathcal{T})$ as follows: if there is $n \in \mathbb{N}$ such that $t \notin d^n(\mathcal{T})$, then $rank(t, \mathcal{T})$ is the smallest n' such that $t \notin d^{n'}(\mathcal{T})$; and ω otherwise. It is obvious that $FCB(\mathcal{T}) \geq k$ if and only if there is $t \in \mathcal{T}$ such that $rank(t, \mathcal{T}) = k$, and for r the root of $\mathcal{T}$, $FCB(\mathcal{T}) = rank(r, \mathcal{T})$. The first simple lemma is the following:

Lemma 13 *Given a tree $\mathcal{T}$, we have $FCB(\mathcal{T}) \geq k + l - 1$, if and only if there is a subtree $\mathcal{S} \subseteq \mathcal{T}$, such that 1. for all leaves s of $\mathcal{S}$, we have $rank(s, \mathcal{T}) = k$, and $FCB(\mathcal{S}) = l$*

Proof. *If*: As the rank of the leaves of $\mathcal{S}$ is k, we have $\mathcal{S} \subseteq d^{k-1}(\mathcal{T})$; as $FCB(\mathcal{S}) = l$, we have $d^{l-1}(d^{k-1}(\mathcal{T})) \neq \mathcal{T}_\emptyset$.

Only if: Assume $FCB(\mathcal{T}) \geq k + l - 1$. Put $\mathcal{S}' = d^{k-1}(\mathcal{T})$. For all leaves s of $\mathcal{S}'$, we have $rank(s, \mathcal{T}) = k$. Moreover, $FCB(\mathcal{S}) \geq l$, because $FCB(\mathcal{T}) \geq k + l - 1$. We thus have to choose an appropriate subtree of $\mathcal{S} \subseteq \mathcal{S}'$. By the subtree properties, this tree exists. □

The next result concerns a property of substitutions. For two (linear) substitutions σ_1, σ_2, define $\sigma_1 \circ \sigma_2$ by $\sigma_1 \circ \sigma_2(a) = \sigma_1(\sigma_2(a))$. $\sigma_1 \circ \sigma_2$ is again a substitution (though not necessarily a linear one, if both σ_1, σ_2 are linear). It is easy to see that for any language L, $\sigma_1[\sigma_2[L]] = \sigma_1 \circ \sigma_2[L]$, for if $w \in \sigma_1 \circ \sigma_2[L]$, then there is $a_1...a_i \in L$, and $w \in \sigma_1 \circ \sigma_2(a_1)...\sigma_1 \circ \sigma_2(a_i)$; consequently, $w \in \sigma_1(\sigma_2(a_1))...\sigma_1(\sigma_2(a))$, and thus $w \in \sigma_1[\sigma_2[L]]$. Conversely, if $w \in \sigma_1[\sigma_2[L]]$, then there is $a_1...a_i \in L$, and $w \in \sigma_1(\sigma_2(a_1))...\sigma_1(\sigma_2(a_i))$. Consequently, $w \in \sigma_1 \circ \sigma_2(a_1)...\sigma_1 \circ \sigma_2(a_i)$, and $w \in \sigma_1 \circ \sigma_2(a_1...a_n) \subseteq \sigma_1 \circ \sigma_2[L]$. From this, we immediately obtain the following:

Corollary 14 *1. $\sigma_1 \circ (\sigma_2 \circ \sigma_3)[L] = (\sigma_1 \circ \sigma_2) \circ \sigma_3(L)$.*
2. $\sigma_1[..[\sigma_k[L]]...] = \sigma_1 \circ ... \circ \sigma_k[L]$.
*3. If σ_1 is a substitution into k-**LIN**, σ_2 into l-**LIN**, then $\sigma_1 \circ \sigma_2$ is a substitution into $k + l$-**LIN**.*

1. and 2. are immediate, 3. follows the easily from 1. and 2. We are now ready for the second main theorem.

[13] Note that 2-**LIN** is already a superclass of the class of languages recognized by k-turn pushdown automata, that is, by PDA which can change from pushing to popping and vice versa at most k times (see [1] for reference). This is because each such language is obtained by a linear substitution on a finite language, and every finite language is linear.

Theorem 15 *There is a CFG G with $L(G) = L$ and $FCB(G) \leq k + 2$, if and only if $L \in k$-**LIN**.*

Proof. *If*: Induction. For $k = 0$ this means: if L is linear, then there is a G with $L(G) = L$ and $FCB(G) \leq 2$. We leave this to the reader. Now assume (induction hypothesis) that for some k, if $L \in k$-**LIN**, then there is a G with $L(G) = L$ and $FCB(G) \leq k+2$. Assume (induction step) $L' \in k+1$-**LIN**. Then there is $L \in k$-**LIN** and a linear substitution σ such that $L' = \sigma[L]$. Furthermore, there is G with $L(G) = L$ and $FCB(G) \leq k + 2$. We can now effect a linear grammar substitution of G simulating σ, thereby obtaining G' with $L(G') = L'$. Now take an arbitrary $\mathcal{T} \in D(G')$. It consists of a tree $\mathcal{S}$ in $D(G)$, with (possibly empty) linear grammar derivation trees departing from its leaves. If we now take $d(\mathcal{T})$, all these linear trees become chains, and as they are attached to leaves of $\mathcal{S}$, we have $FCB(d(\mathcal{T})) = FCB(\mathcal{S})$, and so $FCB(\mathcal{T}) \leq FCB(\mathcal{S}) + 1$, and so $FCB(G') \leq FCB(G) + 1$.

Only if: Induction. The base case is simple: if $FCB(G) \leq 2$, then G is linear (modulo unary branching, which can be eliminated anyway). Now assume (induction hypothesis) that for all grammars G with $FCB(G) \leq k+2$, $L(G) \in k$-**LIN**. Induction step: assume $FCB(G) \leq k + 3$. Then by lemma 13, for each tree $\mathcal{T} \in D(G)$, we have a subtree $\mathcal{S} \subseteq \mathcal{T}$, with $FCB(\mathcal{S}) = 2$, and for each leaf s of $\mathcal{S}$, we have $rank(s, \mathcal{T}) \leq k + 2$. Now for each of these s, the non-terminal corresponding to its label derives a set of trees of FCB-rank $\leq k + 2$, as otherwise it contradicts, with lemma 13, our induction hypothesis. Moreover, by induction hypothesis, the languages derivable from these non-terminals (for N, this is defined as $\{w \in A^* : N \vdash_G^* w\}$) are in k-**LIN**. Call the grammars which result each from making one of these non-terminals the start-symbol of the grammar $G_N : N \in \mathcal{N}$ (however, that does not concern all non-terminals, only those which are only labels of nodes of rank $\leq k + 2$!). This for the "outer trees". The "inner trees" $\mathcal{S}$ of FCB-rank 2 (modulo unary branching) are all linear grammar derivation trees, and if we consider the language formed by their leaves, we get a linear language. We can now construct a grammar which generates exactly the "inner trees" of FCB-rank 2 of the G-derivation trees, such that the leaves, instead of being non-terminals, are terminals being in one-to-one correspondence with non-terminals of G (we write $a_N : N \in \mathcal{N}$). Call this grammar G^i; it is clear that $L(G^i)$ is linear. We can now obtain $L(G)$ as follows: define σ by $\sigma(a_N) = L(G_N)$ (on other letters, let it compute the identity). We now have $\sigma[L(G^i)] = L(G)$, and by construction and corollary 14, $\sigma[L(G^i)]$ is a $k + 1$-linear language, so $L(G) \in k + 1$-**LIN**. □

We have already mentioned the following result of [4]:

Theorem 16 *(Greibach) For each $k \in \mathbb{N}$, k-**LIN** $\subsetneq k + 1$-**LIN**.*

As an immediate consequence, it follows that:

Corollary 17 *$CFL \supsetneq \bigcup_{k\in\mathbb{N}} k$-**LIN**; there are context-free languages L such that $L \notin k$-**LIN** for all $k \in \mathbb{N}$.*

Proof. Assume the contrary: for some $k \in \mathbb{N}$, k-**LIN** $\supseteq CFL$. As for all $n \in \mathbb{N}$, n-**LIN** $\subseteq CFL$, that means that k-**LIN** $\supseteq (k+1)$-**LIN**, contradiction. □

So the classes of languages generated by CFG of FCB-rank k also form a proper, infinite hierarchy; furthermore, there are CFLs for which no grammar of bounded FCB rank exists:

Corollary 18 *For every $k \in \mathbb{N}$, there is a language L such that there is a CFG G with $FCB(G) = k+1$ and $L(G) = L$, but no CFG G' with $FCB(G') \leq k$ and $L(G') = L$. Moreover, there is a CFL L, such that if $L(G) = L$, it follows that $FCB(G) = \omega$.*

The typical candidate for CFLs which are not in k-**LIN** is the family of Dyk-languages over alphabets of arbitrary cardinality. We do not have to explain that inverse implications of the form: "if $FCB(G) > k$ or $FCB(G) = \omega$, then $L(G)$ is X" are not legitimate: as is well-known, it is undecidable whether a CFG generates a regular or linear language. So knowing the FCB rank of a CFG does not help us in establishing a *lower bound* of the complexity of its language, it only might give an upper bound.

7 Beyond Context-free Grammars

We have seen that a bounded FCB-rank for CFG corresponds to a well-established class of languages. The interesting thing is that our treatment is by no means restricted to CFG, as trees arise in many ways in formal language theory: in particular, we can look at formalisms beyond CFG. There are two primary ways to go: firstly, there are formalisms which keep the rule format of context-free grammars, while allowing operations on strings which are more complex than just concatenation. The most prominent example for this are multiple context-free grammars (MCFG,[11]). On the other side, there are formalisms extending rule-schemes directly to trees, such as regular tree grammars (RTG), tree-adjoining grammars (TAG), and more generally, context-free tree grammars (CFTG) (see [9], [2]). Regarding formalisms as MCFG, there is little of interest we can say: derivation trees are usually defined in exactly the same manner as in CFG. In this sense all tree-based results for CFG transfer to MCFG. What changes of course are the language-theoretic properties, but those are not the results we are interested in in the first place. So we will use the rest of this paper to show how the FCB-rank works for tree grammars.

We have mentioned RTG, TAG and CFTG. For reasons of space, we will only define and look at (a particular class of) CFTG, because the two former can be seen as special cases of the latter, and so they are *a fortiori* covered by our treatment.[14] The foundation for tree grammars is the representation of finite, ordered, labelled trees as terms. Let Σ be an alphabet. $term(\Sigma)$ is defined as the

[14] More precisely, an RTG is a CFTG where all non-terminals have arity 0, and TAG correspond to simple monadic CFTG, see definitions below.

smallest set such that 1. if $a \in \Sigma$, then $a \in \mathit{term}(\Sigma)$, and if $\mathtt{t}_1, ..., \mathtt{t}_i \in \mathit{term}(\Sigma)$, $a \in \Sigma$, then $a(\mathtt{t}_1, ..., \mathtt{t}_i) \in \mathit{term}(\Sigma)$. Let X be a countable set of variables. $\mathit{term}_X(\Sigma)$ is defined as follows: if $a \in \Sigma$, then $a \in \mathit{term}_X(\Sigma)$; if $x \in X$, then $x \in \mathit{term}_X(\Sigma)$; if $\mathtt{t}_1, ..., \mathtt{t}_i \in \mathit{term}_X(\Sigma)$, $a \in \Sigma$, then $a(\mathtt{t}_1, ..., \mathtt{t}_i) \in \mathit{term}_X(\Sigma)$. We generally use $\mathtt{t}$ as meta-variable for terms. By $\mathtt{t}[x_1, ..., x_i]$ we intend a term, such that the variables occurring therein are among $\{x_1, ...x_i\}$. It is easy to see how to translate terms into finite, ordered, labelled trees (up to isomorphism). A problem is that a term does not specify the names of nodes, yet we need some address to identify nodes. We therefore assume that in trees corresponding to terms, nodes are named as in a **tree domain**. A tree domain is a set $T \subseteq \mathbb{N}^*$ (we think here of numbers as abstract entities, not in binary, ...,decimal etc. representation), such that for $n, m \in \mathbb{N}$, $\overline{n}, \overline{m} \in \mathbb{N}^*$, if $\overline{nm} \in T$, then $\overline{n} \in T$, and if $\overline{n}n\overline{m} \in T$, $m < n$, then $\overline{n}m \in T$. Given a tree domain, we define the precedence order $\preceq$ as the (reflexive) lexicographic order *lex*, and $\trianglelefteq$ as the prefix relation *pref*. Tree domains give us a "canonical" node labels for any (ordered) tree, and so for $\mathtt{t} \in \mathit{term}_X(\Sigma)$, we denote by $\hat{\mathtt{t}}$ its associated canonical tree. In the sequel, we will sometimes mix notions of trees and notions of terms; this will usually not lead to confusion and keep the treatment to a manageable size. Contrary to subtrees, subterms are generally thought to be contingent: if $\mathtt{t} = a \in \Sigma$, then $\mathit{subterm}(\mathtt{t}) = \{a\}$; if $\mathtt{t} = a(\mathtt{t}_1, \dots, \mathtt{t}_i)$, then $\mathit{subterm}(\mathtt{t}) = \{\mathtt{t}\} \cup \bigcup_{j=1}^{i} \mathit{subterm}(\mathtt{t}_i)$. We write $\mathtt{t}[\mathtt{t}_1, \dots, \mathtt{t}_i]$ if $\mathtt{t}_1, \dots, \mathtt{t}_i \in \mathit{subterm}(\mathtt{t})$. For $\sigma_1, \dots, \sigma_i \in \Sigma$, we write $\mathtt{t}\langle\sigma_1, ..., \sigma_i\rangle$, if there are nodes $t_1, ..., t_i$ in $\hat{\mathtt{t}}$ labelled with $\sigma_1, ..., \sigma_i$, respectively, and for no two distinct $t_n, t_m \in \{t_1, ..., t_i\}$, we have $t_n \trianglelefteq t_m$. Let $\mathtt{t}[x_1, \dots, x_i] \in \mathit{term}_X(\Sigma)$ with $x_1, \dots, x_i \in X$. By $\mathtt{t}[\mathtt{t}_1/x_1, \dots, \mathtt{t}_i/x_i]$ we denote the term which results by substituting $\mathtt{t}_1$ for x_1, ..., $\mathtt{t}_1$ for x_1.

A **CFTG** is a tuple $(\mathtt{S}, \mathcal{N}, \Sigma, R)$, where $\mathcal{N}$ and Σ are disjoint, finite sets (of terminals and non-terminals), $\mathtt{S} \in \mathcal{N}$, and $R \subseteq (\mathit{term}_X(\Sigma \cup \mathcal{N}))^2$, where all rules in R have the form $(N(x_1, ..., x_i), \mathtt{t}[x_1, ..., x_i])$, by which we hereby include the possibility that $x = 0$, and thus $N(x_1, ..., x_i) = N$. In general, we assume that non-terminals have a fixed arity, that is, they always occur with the same number of variables/subterms in R. The derivability relation $\vdash_G \subseteq (\mathit{term}_x(\Sigma \cup \mathcal{N}))^2$, for G a CFTG, is defined as follows. We have $\mathtt{t}[N(\mathtt{t}_1, ..., \mathtt{t}_i)] \vdash_G \mathtt{t}[\mathtt{t}'[\mathtt{t}_1/x_1, ..., \mathtt{t}_i/x_i]]$, iff $(N(x_1, ..., x_i), \mathtt{t}'[x_1, ..., x_i]) \in R$; $\vdash_G^*$ is the reflexive and transitive closure, and $L(G) = \{\mathtt{t} \in \mathit{term}(\Sigma) : \mathtt{S} \vdash_G^* t\}$. We now define OI-derivations: We write $\mathtt{t}[N(\mathtt{t}_1, ..., \mathtt{t}_i)] \vdash_{GOI} \mathtt{t}[\mathtt{t}'[\mathtt{t}_1/x_1, ..., \mathtt{t}_i/x_i]]$, if in $\hat{\mathtt{t}}[N(\mathtt{t}_1, ..., \mathtt{t}_i)]$, there is no node labelled by a non-terminal which dominates (a node labelled by) the non-terminal we expand in the derivation. $\vdash_{GOI}^*$ is the reflexive and transitive closure of $\vdash_{GOI}$. For $\mathtt{t} \in \mathit{term}_X(\Sigma)$, we have $\mathtt{S} \vdash_G^* \mathtt{t}$ if and only if $\mathtt{S} \vdash_{GOI}^* \mathtt{t}$; this is an important fact of which we will make use in the sequel.

A CFTG is said to be **linear**, if for $(N(x_1, ..., x_i), \mathtt{t}[x_1, ..., x_i]) \in R$, each $x_1, ..., x_i$ occurs at most once in $\mathtt{t}[x_1, ..., x_i]$; it is **non-deleting**, if each variable occurs at least once; grammars which are both linear and non-deleting are said to be **simple**. To simplify our treatment, we will assume that our grammars are simple. Note that this restriction does not come without loss of generality (see [6]). But it is also well-known that linear and non-deleting CFTG still have considerable expressive power: for example, TAG are equivalent to CFTG which are simple and allow at most one variable to occur on each side of a rule (modulo some details, see [7]). Simple CFTG are particularly interesting, because the languages they generate are semilinear. Note that for $\mathtt{t} \vdash_G^* \mathtt{t}'$, we also allow variables to occur in $\mathtt{t}, \mathtt{t}'$, but by definition of $\vdash_G^*$ and assumption of non-deleting rules, a variable x then occurs in $\mathtt{t}$ if and only if it occurs in $\mathtt{t}'$. Furthermore, we will assume (this time without loss of generality) that all nonterminals of our CFTG are both reachable and groundable. FCB-rank of a CFTG is defined in a straightforward fashion by $FCB(G) := sup\{FCB(\hat{\mathtt{t}}) : \mathtt{t} \in L(G)\}$. Note that we do not use derivation trees, but the trees generated by the grammar. As we do not generate infinite trees, we cannot use maximum instead of supremum, so we have to consider this in our constructions.

By $yield : term(\Sigma) \to \Sigma^*$ we denote the function mapping trees to words by forming the concatenation of their leaves; it is extended to sets in the canonical fashion. The above mentioned correlation of TAG and simple, monadic CFTG entails that we get a CFTG G with $FCB(G) = 2$, where $yield(L(G))$ is not context-free. Consider the rules:

$$\mathtt{S} \to e(a, N(e(b, c)), d);\ N(x) \to e(a, N(e(b, x, c)), d);\ N(x) \to e(x).$$

This yields $\{a^n b^n c^n d^n : n \in \mathbb{N}\}$; yet, all derived trees have rank 2.

8 FCB-Rank of CFTG

We will now show that it is decidable whether $FCB(G) = \omega$ for a simple CFTG G. The procedure is more complicated than for CFG, so we will show its steps separately. A particularly useful normal-form for CFTG would be one where all trees on the right-hand side of rules have depth 2; this however is not available for the general case we consider. We will however assume without loss of generality that CFTG-rules are in **non-terminal normal form** (NTNF): all right-hand sides of rules have the form $\tau(\mathtt{t}_1, ..., \mathtt{t}_i)$, where $\tau \in \mathcal{N} \cup \Sigma$, and $\mathtt{t}_1, .., \mathtt{t}_i \in term_X(\mathcal{N})$. We thus do not allow terminals on the right hand side, except for the root of term-trees. It is easy to see that any CFTG can be brought into nonterminal normal form without substantial modification. Recall that we assume that all non-terminals are reachable and groundable.

Lemma 19 *Let G be a simple CFTG. $FCB(G) = \omega$, if and only if there is an $N \in \mathcal{N}$, such that $N(x_1, ..., x_i) \vdash_G^* \mathtt{t}\langle N, N\rangle$.*

Proof. *If*: Assume we have such an $N \in \mathcal{N}$. Then $\mathtt{S} \vdash_{GOI}^* \mathtt{t}[N(\mathtt{t}_1, ..., \mathtt{t}_i)]$, where the node labelled by N is not dominated by any nonterminal. As its position will not change in the final derived tree, we can already give him its address in the tree domain, say α. By assumption, $N \vdash_{GOI}^* t\langle N, N\rangle$, where none of the two nodes labelled N is dominated by any non-terminal; so having the address α_1, α_2 in $t\langle N, N\rangle$, we have $\mathtt{t}[N(\mathtt{t}_1, ..., \mathtt{t}_i)] \vdash_{GOI}^* \mathtt{t}'\langle N, N\rangle$, where the two nodes labelled N have address $\alpha\alpha_1, \alpha\alpha_2$. Iterating this argument, we can generate a tree with an arbitrarily large finite complete binary subtree.

Only if: Assume there is no such $N \in \mathcal{N}$. The argument we use is similar to the CFG case, though slightly more delicate. If S derives a tree of arbitrarily large FCB-rank, then it follows that there must be N_1, N_2, such that $S \vdash_G^* t\langle N_1, N_2\rangle$, none of the two is dominated by another non-terminal, and both N_1, N_2 derive trees of arbitrarily large FCB-rank (this uses OI-derivation and NTNF); otherwise, we run into a contradiction.[15] Now by assumption, only one of N_1, N_2 can derive S; assume without loss of generality it is N_1. For N_2 it also holds that there must be N_{21}, N_{22}, such that $N_2 \vdash_G^* t\langle N_{21}, N_{22}\rangle$, none of the two is dominated by another non-terminal, and both N_1, N_2 derive trees of arbitrarily large FCB-rank; none of them can derive S, and only one of them N_2. Pick the other one etc. Iterating this, we finish up with a non-terminal M, such that M can derive trees of arbitrarily large FCB-rank, yet it cannot introduce any non-terminals - contradiction. □

Lemma 20 *For a CFTG G, $N \in \mathcal{N}$, it is decidable whether $N \vdash_G^* \mathtt{t}\langle N, N\rangle$ for some* $\mathtt{t}$.

Proof. We start by an algorithm checking whether $N \vdash_G^* \mathtt{t}\langle M\rangle$. That is easily decidable: just check whether $N \vdash_G \mathtt{t}\langle M\rangle$; and next check for all N' : $N \vdash_G \mathtt{t}\langle N'\rangle$ (except N) whether $N' \vdash_G \mathtt{t}\langle M\rangle$ etc. The procedure for checking $N \vdash_G^* \mathtt{t}\langle N, N\rangle$ is based on this: we check whether $N \vdash_G \mathtt{t}\langle N, N\rangle$, and then check for all M, M' such that $N \vdash_G \mathtt{t}\langle M, M'\rangle$ whether $M, M' \vdash_G^* \mathtt{t}\langle N\rangle$, $M \vdash_G^* \mathtt{t}\langle N, N\rangle$ or $M' \vdash_G^* \mathtt{t}\langle N, N\rangle$ and iterate this until we have checked immediate derivability for all non-terminals. □

Corollary 21 *Given a simple CFTG G, we can decide whether $FCB(G) = \omega$ or $FCB(G) < \omega$.*

So we have decidability for the infinite case. How about the finite case? Here are several problems to overcome; for reasons of space, we do not present the algorithm we think does the job, but only put forward the following conjecture:

Conjecture 22 *Given a simple CFT G, $FCB(G) < \omega$, there is a terminating algorithm which computes $FCB(G)$.*

The proof of this is based on n-cuts of trees: intuitively, an n-cut of a tree is a subset of its domain, such that its $\trianglelefteq$-predecessors form a subtree with FCB-rank n (thus an n-cut must contain at least 2^{n-1} elements). We can use these n-cuts to determine how CFTG-rules increase the rank of derived trees. Important open questions are the following: what classes of (string) languages correspond with CFTG with (a particular) bounded FCB-rank? It is clear that they are not contained in the context-free languages; do they conversely contain the context-free languages for a certain rank?

[15] Actually, this uses still another fact: if we substitute trees of finite FCB-rank for the nodes of a tree of finite FCB-rank, we get a tree of finite FCB-rank. This argument is rather simple, though, and we do not have the space to introduce the relevant notions.

9 Conclusions

The main goal of this paper was to introduce the notion of finitary Cantor-Bendixson rank of trees, and establish its major properties. The first main result was that the rank of a context-free grammar is decidable. This is surprising, because many similar properties of CFGs are undecidable. The explanation is that it is a property concerning the strong generative capacity of a CFG, rather than the language generated. Our second main result was that there is a correspondence of two proper infinite hierarchies, the hierarchy of k-linear languages, and the class of grammars with FCB-rank $k + 2$. This is an interesting result, because the former notion is languages-theoretic, whereas the latter comes from relation-theory, and when applied to grammars, refers to their strong generative capacity. We thus have an interesting relation between notions from very diverse fields. The third main result concerned the decidability of the FCB-rank of simple context-free tree grammars. We have shown that it is decidable whether a simple CFTG has infinite rank and put forward the conjecture that its precise rank is computable; for reasons of space, we could not present the algorithm for its computation and its correctness proof.

References

1. Jean Berstel. *Transductions and Context-free Languages.* Teubner, Stuttgart, 1979.
2. Joost Engelfriet and Erik Meineche Schmidt. Io and oi. i. *J. Comput. Syst. Sci.*, 15(3):328–353, 1977.
3. Roland Fraïssé. *Theory of Relations.* Studies in logic and the foundations of mathematics ; 118. North-Holland, 1986.
4. Sheila A. Greibach. Chains of full AFL's. *Mathematical Systems Theory*, 4:231–242, 1970.
5. John A. Hawkins. *Efficiency and Complexity in Grammars.* Oxford Univ. Pr., Oxford, 2004.
6. Dieter Hofbauer, Maria Huber, and Gregory Kucherov. Some results on top-context-free tree languages. In Sophie Tison, editor, *CAAP*, volume 787 of *Lecture Notes in Computer Science*, pages 157–171. Springer, 1994.
7. Stephan Kepser and Jim Rogers. The equivalence of tree adjoining grammars and monadic linear context-free tree grammars. *Journal of Logic, Language and Information*, 20(3):361–384, 2011.
8. Marcus Kracht. *The Mathematics of Language.* Number 63 in Studies in Generative Grammar. Mouton de Gruyter, Berlin, 2003.
9. William C. Rounds. Mappings and grammars on trees. *Mathematical Systems Theory*, 4(3):257–287, 1970.
10. Sasha Rubin. Automata presenting structures: A survey of the finite string case. *Bulletin of Symbolic Logic*, 14(2):169–209, 2008.
11. Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context–free grammars. *Theor. Comp. Sci.*, 88:191–229, 1991.
12. Harold Simmons. The extendend cantor-bendixson analysis of trees. *Algebra Universalis*, 52:439–468, 2005.