Sequential Grammars and Automata with Valences

Henning Fernau and Ralf Stiebe

WSI-2000-25

Henning Fernau
Wilhelm-Schickard-Institut
für Informatik
Universität Tübingen
Sand 13
D-72076 Tübingen
Germany

email:`fernau@informatik.uni-tuebingen.de`
Telefon: (07071) 29-77565
Telefax: (07071) 29-5061

Ralf Stiebe
Institut für Informatik
Martin-Luther-Universität
Halle-Wittenberg
Kurt-Mothes-Str. 1
D-06120 Halle
Germany

`stiebe@informatik.uni-halle.de`
Telefon: (0345)552-4736
Telefax: (0345)552-7009

ISSN 0946-3852

Sequential Grammars and Automata with Valences

Henning Fernau
Wilhelm-Schickhard-Institut für Informatik
Universität Tübingen
Sand 13, D-72076 Tübingen, Germany
email: fernau@informatik.uni-tuebingen.de

Ralf Stiebe
Institut für Informatik
Martin-Luther-Universität Halle-Wittenberg
Kurt-Mothes-Str. 1, D-06120 Halle, Germany
email: stiebe@informatik.uni-halle.de

December 22, 2000

Abstract

We discuss the model of valence grammars, a simple extension of context-free grammars. We show closure properties of valence languages over arbitrary monoids. Chomsky and Greibach normal form theorems and an iteration lemma for valence grammars over the groups $\mathbb{Z}^k$ are proved. The generative power of different control monoids is investigated. In particular, we show that valence grammars over finite monoids or commutative monoids have the same power as valence grammars over finite groups or commutative groups, respectively.

1 Introduction

Valences were introduced in 1980 by Păun [16] as a mechanism of regulated rewriting. The original idea was to assign to a context-free core rule an integer, the so-called valence, and to compute for a derivation a value by adding all the valences of the applied rules. A derivation is valid iff this sum evaluates to 0, reflecting the balance of positive and negative valences in chemical molecules or in directed graphs. This mechanism can be easily extended to monoids different from $(\mathbb{Z}, +, 0)$.

Valence languages have been in the focus of several papers [8, 13, 14, 17, 18, 25, 26]. We think for several reasons that valence grammars are worth a deeper investigation. First of all, valences are a very natural and simple mechanism. The context-free derivation process is not changed at all; the validity of a derivation is only checked at the end. Thus, many attractive properties of context-free grammars can be immediately transferred. Moreover, it is possible to describe several language families by valence grammars over different monoids, and one can hope to simplify investigations concerning these families by studying the corresponding valence grammars. For example, unordered vector languages can be characterized via valence languages over the monoid of positive rational numbers with multiplication, and Greibach's family *BLIND* of languages accepted by blind multi-counter automata can be generated by regular valence grammars over $(\mathbb{Q}_+, \cdot, 1)$. Finally, valences are very flexible and can be incorporated into parallel systems, grammars with other means of regulation and machine models (in fact, "finite valence automata" were discussed by several authors even before the introduction of valence grammars, e.g., in [9, 12, 19]).

This paper discusses valences in sequential context-free grammars. Valences in parallel systems and in combination with other modes of regulation are considered in separate papers [6, 7]. The necessary notations are given in Section 2.

Valence automata and transducers are discussed in Section 3. In Section 4, we discuss valence languages over various monoids. It is shown that context-free and regular languages over arbitrary monoids are semi-AFL's. The known results regarding closure properties for specific monoids are extended and generalized, while the proofs are simplified. The concept of a derivation tree is generalized. As regards the generative power of valence grammars, in Section 4.3, we show that valence grammars over finite monoids or commutative monoids have the same power as valence grammars over finite groups or commutative groups, respectively. Then, we concentrate on valence grammars over the monoids $(\mathbb{Z}^k, +, \vec{0})$ and $(\mathbb{Q}_+, \cdot, 1)$. In Section 5, we show how to construct (Chomsky and Greibach) normal forms for these valence grammars, a result which also applies to the equivalent class of unordered vector grammars. An iteration lemma for the same control monoids is given in Section 6.

Remark: Some results of this paper appeared in an extended abstract which was published within the proceedings of the 22nd MFCS conference 1997, see [5]. There, results on parallel grammars with valences were also shown. For a long version of these results, the reader is referred to [6].

2 Preliminaries

Throughout the paper, we assume the reader to be familiar with the theory of context-free languages, see, e.g., [11, 22]. Moreover, some familiarity with basic algebraic notions is helpful.

Firstly, we recall some algebraic notions. A semigroup is a set together with a binary, associative operation on it. A semigroup with a neutral element is called monoid. Formally, a monoid $\mathbf{M}$ can be specified as $\mathbf{M} = (M, \circ, e)$, where M is the underlying set of the monoid, $\circ$ is the binary operation of the monoid, and e is its neutral element. A monoid in which every element a possesses an inverse a^{-1} is called a group. A semigroup (homo)morphism is a mapping of one semigroup into another one which respects the two involved semigroup operations. A bijective morphism is called isomorphism. If A is the subset of a semigroup S, $\langle A\rangle$ denotes the subsemigroup generated by A, i.e., the smallest subsemigroup of S containing A. Similar notions can be introduced for monoids and groups (instead of semigroups).

$\mathbb{N}$ is the set of natural numbers including 0. $\mathbb{N}_+ = \mathbb{N}\setminus\{0\}$ is the set of positive integers. $\mathbb{Z}$ is the set of integers. $\mathbb{Q}_+$ denotes the set of positive rational numbers.

The monoids $(\mathbb{Z}^k, +, \vec{0})$ and $(\mathbb{Q}_+, \cdot, 1)$ are sometimes simply denoted by $\mathbb{Z}^k$ and $\mathbb{Q}_+$. The canonical basis vectors of $\mathbb{Z}^k$ are written $\vec{e_i}$, $1 \leq i \leq k$, i.e., all components of $\vec{e_i}$ are zero except for the ith component which equals one. For a vector $\vec{r} = (r_1, r_2 \dots, r_k) \in \mathbb{Z}^k$, we define the max-norm by $||\vec{r}||_{\max} = \max\{|r_i| : 1 \leq i \leq k\}$ and the 1-norm by $||\vec{r}||_1 = \sum_{i=1}^{k} |r_i|$. We define the modulo and integer division operations for vectors, $\operatorname{mod}, \operatorname{div} : \mathbb{Z}^k \times \mathbb{Z} \to \mathbb{Z}^k$, as the component-wise application of the integer operations $\operatorname{mod}, \operatorname{div} : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$, and denote them by $\vec{r} \mod m$ and $\vec{r} \operatorname{div} m$, for $\vec{r} \in \mathbb{Z}^k, m \in \mathbb{Z}$.

If R is some binary relation, R^+ denotes the transitive closure of R and R^* the transitive reflexive closure of R. The inclusion relation is denoted by $\subseteq$, proper inclusion by $\subset$.

Now, we recall some formal language notions. Let $V = \{a_1, \dots, a_n\}$, $n \geq 1$, be an alphabet. The set of all words over V is denoted by V^*, the empty word by λ, and $V^+ = V^* \setminus \{\lambda\}$. Together with the concatenation operation, V^+ forms a semigroup, and V^* is a monoid with λ as its neutral element. For $w \in V^*$, the length of w is denoted by $|w|$, the number of appearances of the letter $a \in V$ in w is denoted by $|w|_a$. The *Parikh mapping* associated with V is a map $\Psi : V^* \to \mathbb{N}^n$ such that $\Psi(w) = (|w|_{a_1}, \dots, |w|_{a_n})$. For a language $L \subseteq V^*$, we define the *Parikh set of* L by $\Psi(L) = \{\Psi(w) : w \in L\}$. Two languages $L_1, L_2 \subseteq V^*$ are called *letter equivalent* iff their Parikh sets are equal. For a word w, let $\operatorname{Perm}(w)$ denote the set of all words obtained by permuting the symbols of w. For a language L, we define $\operatorname{Perm}(L) = \bigcup_{w\in L} \operatorname{Perm}(w)$.

A *context-free grammar* is a quadruple $G = (N, T, P, S)$, consisting of a nonterminal alphabet N, a terminal alphabet T, $N \cap T = \emptyset$, a set of rules $P \subseteq N \times (N \cup T)^*$, and a start symbol S. A string $\alpha \in (N \cup T)^*$ directly derives the string $\beta \in (N \cup T)^*$, denoted as $\alpha \Rightarrow \beta$, iff there is a rule $A \rightarrow \gamma$ in P such that $\alpha = \alpha_1 A \alpha_2$ and $\beta = \alpha_1 \gamma \alpha_2$. The language generated by G is $L(G) = \{w \in T^* : S \overset{*}{\Rightarrow} w\}$, where $\overset{*}{\Rightarrow}$ denotes the reflexive and transitive closure of $\Rightarrow$. For a derivation Δ in G which applies the rules $p_1, p_2, \ldots, p_n$ (in this sequence), the *control word* of Δ is defined as $c(\Delta) = p_1 p_2 \ldots p_n$.

Finally, we define the central concept of this paper. A *(context-free) valence grammar* over the monoid $\mathbf{M} = (M, \circ, e)$ is a construct $G = (N, T, P, S, \mathbf{M})$, where N, T, S are defined as in a context-free grammar, i.e., N is the alphabet of nonterminals, T (with $T \cap N = \emptyset$) is the alphabet of terminals, $S \in N$ is the start symbol, and $P \subseteq N \times (N \cup T)^* \times M$ is a finite set of *valence rules*. For a valence rule $p = (A \rightarrow \alpha, m)$, the rule $A \rightarrow \alpha$ is called the *core rule* of p, while m is called the *valence* of p. The function $\mathrm{val} : P \rightarrow M$, mapping a valence rule to its valence, is called the valence mapping (which can be extended to a monoid morphism from P^* to M). To avoid explicit reference to the monoid, we write $\mathrm{Lab}(G)$ for the set of all valences appearing in P, instead of $\mathrm{val}(P)$. The yield relation $\Rightarrow$ over $(N \cup T)^* \times M$ is defined as: $(w, m) \Rightarrow (w', m')$ iff there is a rule $(A \rightarrow \alpha, n)$ such that $w = w_1 A w_2$, $w' = w_1 \alpha w_2$ and $m' = m \circ n$. The language generated by G is $L(G) = \{w \in T^* : (S, e) \overset{*}{\Rightarrow} (w, e)\}$.

A valence grammar is called *regular* or, more specifically, *right-linear* if all its core rules are right-linear, i.e., they are all of the form $A \rightarrow wB$ with $A \in N$, $w \in T^*$ and $B \in N \cup \{\lambda\}$; a valence grammar is λ-free if it has no core rule of the form $A \rightarrow \lambda$. The language families generated by context-free, context-free λ-free and regular valence grammars over $\mathbf{M}$, are denoted by $\mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbf{M})$, $\mathcal{L}(\mathrm{Val}, \mathrm{CF} - \lambda, \mathbf{M})$ and $\mathcal{L}(\mathrm{Val}, \mathrm{REG}, \mathbf{M})$, respectively. For brevity, let $\mathbb{Z}^0$ denote the trivial monoid. Then, $\mathcal{L}(\mathrm{Val}, X, \mathbb{Z}^0) = \mathcal{L}(X)$ for $X \in \{\mathrm{REG}, \mathrm{CF} - \lambda, \mathrm{CF}\}$.

In terms of control words, a derivation in the underlying context-free grammar is valid in a valence grammar iff its control word is mapped by the valence morphism to the neutral element. Next, we define some other regulation mechanisms depending on control words and, hence, related to valence grammars. A *matrix grammar* is a quintuple $G = (N, T, P, S, M)$, where $G' = (N, T, P, S)$ is a context-free grammar and $M \subset P^*$ is a finite set of matrices. A terminal derivation Δ in G' is valid in G iff $c(\Delta) \in M^*$. $L(G)$ consists of all words obtained by valid derivations. An *unordered vector grammar* is defined like a matrix grammar, with the difference that a terminal derivation Δ in G' is valid in G iff $c(\Delta) \in \mathrm{Perm}(M^*)$. The families of unordered vector languages of type $X \in \{\mathrm{REG}, \mathrm{CF} - \lambda, \mathrm{CF}\}$ are denoted by $\mathcal{L}(\mathrm{UV}, X)$.

An important tool for proving closure properties is the notion of a finite transducer, which is defined next. A *(finite-state) transducer* is a sextuple

$$\mathcal{A} = (Z, I, O, z_0, \delta, z_f),$$

consisting of the finite set of states Z, the input and output alphabets I and O, the initial state $z_0 \in Z$, the finite transition relation $\delta \subset I^* \times Z \times Z \times O^*$, and the final state $z_f \in Z$. $\mathcal{A}$ is called λ-free iff $\delta \subset I^* \times Z \times Z \times O^+$.

The yield relation $\models$ over $I^* \times Z \times O^*$ is defined as: $(w_{in}, z, w_{out}) \models (w'_{in}, z', w'_{out})$ iff, for some $x \in I^*$ and $u \in O^*$, $w_{in} = xw'_{in}$, $w'_{out} = w_{out}u$ and $(x, z, z', u) \in \delta$.

For $w \in I^*$, the transduced image of w under $\mathcal{A}$ is defined as

$$\tau_{\mathcal{A}}(w) = \{w' \in O^* : (w, z_0, \lambda) \models^* (\lambda, z_f, w')\}.$$

For a language $L \subseteq I^*$, the transduced image is $\tau_{\mathcal{A}}(L) = \bigcup_{w \in L} \tau(w)$. The operator $\tau_{\mathcal{A}}$ on languages is also called *(rational) transduction.*

We briefly mention two well-known facts on finite-state transducers:

1. A language family is a full trio (or trio, respectively) iff it is closed under rational transductions (or λ-free rational transductions, respectively).

2. Every rational transduction can be defined by a finite-state transducer in *normal form*, i.e., with transition relation $\delta \subset (I \cup \{\lambda\}) \times Z \times Z \times (O \cup \{\lambda\})$.

Let us further mention (see [4]) that every λ-free rational transduction is representable as the composition $\tau = \tau_2\tau_1$ of a transduction τ_1, given by a λ-free "normal form" transducer $\mathcal{A}_1$, followed by a restricted erasing τ_2. Recall that a *k-restricted erasing* is a rational transduction τ which realizes the morphism $g_X : (T \cup \{\$\})^* \to X^*$ (where $\$ \notin X$), given by $a \mapsto a$ for $a \in X$ and $\$ \mapsto \lambda$, on the domain

$$\mathrm{dom}(\tau) = (\bigcup_{i=0}^{k} \{\$^i\} X)^+ .$$

3 Valence automata

In analogy to valence grammars, one can define finite valence automata, (finite-state) valence transducers, and valence pushdown automata where, for each such automaton, a valence is assigned to each transition, and a run of the automaton is valid iff the valence product evaluates to the neutral element. The family of languages accepted by nondeterministic finite valence automata over $\mathbf{M}$ (possibly with λ-moves) is denoted by $\mathcal{L}(\mathrm{Val}, \mathrm{NFA}, \mathbf{M})$. The family of languages accepted by nondeterministic valence pushdown automata over $\mathbf{M}$ (possibly with λ-moves)is denoted by $\mathcal{L}(\mathrm{Val}, \mathrm{NPDA}, \mathbf{M})$. The main purpose of this section is to list relations between valence automata and other kinds of enhanced automata. Moreover, we show that several interesting operations are valence transductions, mostly over $\mathbb{Z}^k$.

Finite valence automata over $\mathbb{Q}_+$ have been investigated as *one-way finite automata with multiplication without equality* by Ibarra, Sahni, and Kim [12].

Blind k-counter machines studied by Greibach [9] are equivalent to finite valence automata over $\mathbb{Z}^k$. An interesting generalization considered in that paper is the notion of a *partially blind k-counter automaton*, where no component is allowed to reach a negative value during a run.

Valence automata over semigroups, with a slightly different acceptance condition (namely, accepting with a finite set or with the homomorphic image of a regular language instead of accepting with the neutral element), were discussed by Red'ko and Lisovik [19]. The most remarkable results are:

- a characterization of the context-free languages by finite valence automata over F_2 and
- a characterization of the recursively enumerable languages by finite valence automata over $F_2 \times F_2$,

where F_2 denotes the free group generated by two elements.

We continue this section by mentioning some interesting properties of valence transductions. The proofs of the propositions are left to the reader, see also [6].

Proposition 3.1 *Let X be an alphabet with k letters, and let $\Psi : X^* \to \mathbb{Z}^k$ be a Parikh mapping. The relation Perm $:= \{(v, w) : \Psi(v) = \Psi(w)\}$ is a valence transduction over $\mathbb{Z}^k$.*

Proposition 3.2 *The operation "intersection with languages from $\mathcal{L}(\mathrm{Val}, \mathrm{NFA}, \mathbf{M})$" is a valence transduction over $\mathbf{M}$ for each monoid $\mathbf{M}$.*

Proposition 3.3 *If τ is a valence transduction over $\mathbf{M}$, then τ^{-1} is a valence transduction over $\mathbf{M}$, as well.*

Proposition 3.4 *Let X be an alphabet with k letters, let $\Psi : X^* \to \mathbb{N}^k$ be a Parikh mapping, and let $S \subseteq \mathbb{N}^k$ be a semilinear set. Then,*

$$\Psi^{-1}(S) := \{v \in X^* : \Psi(v) \in S\} \in \mathcal{L}(\mathrm{Val}, \mathrm{NFA}, \mathbb{Z}^k).$$

Theorem 3.5 *Let $\mathbf{M}, \mathbf{M}'$ be monoids. Then, $L \in \mathcal{L}(\mathrm{Val}, \mathrm{REG}, \mathbf{M} \times \mathbf{M}')$ iff there are a language $L' \in \mathcal{L}(\mathrm{Val}, \mathrm{REG}, \mathbf{M}')$ and a valence transduction τ over $\mathbf{M}$ such that $L = \tau(L')$.*

Proof. Let $L \in \mathcal{L}(\mathrm{Val}, \mathrm{REG}, \mathbf{M} \times \mathbf{M}')$. Then, $L \subseteq T^*$ is generated by a right-linear valence grammar G over the product monoid $\mathbf{M} \times \mathbf{M}'$. We have to construct a right-linear valence grammar G' over the monoid $\mathbf{M}'$ and a valence transduction τ over $\mathbf{M}$ such that $L = \tau(L(G'))$. Let $X \subset \mathbf{M}$ be the monoid elements occurring in rules of G plus the neutral element e. Then, let $(T \cup \{\Lambda\}) \times X$ be the terminal alphabet of G', as well as the input alphabet of τ. For every rule $(A \to \alpha B, (m, m'))$ in G with $\alpha \in T^*$, $A \in N$ and $B \in N \cup \{\lambda\}$, where N is the nonterminal alphabet of G, we put a rule $(A \to \phi(\alpha, m')B, m)$ into G, where $\phi(\lambda, m') = (\Lambda, m') \in (T \cup \{\Lambda\}) \times X$, and $\phi(a_1 \dots a_j, m') = (a_1, m')(a_2, e) \dots (a_j, e) \in ((T \cup \{\Lambda\}) \times X)^+$ with $a_1, \dots, a_j \in T$. Now, consider a valence transduction with a single state [1] which basically maps (Λ, m') into λ and (a, m') into a (with $a \in T$), taking m' into account by means of a corresponding valence of the transduction. Thus, $L = \tau(L(G'))$ as required.

The other direction is quite similar to the classical triple construction for showing closure of the regular languages under transductions and is, hence, left to the reader. □

Remark 3.6 *A result similar to the preceding theorem can be proved for valence* NPDA*'s instead of regular valence grammars.*

In the case of context-free valence grammars, $\mathbf{M}$ *has to be commutative. Moreover, in the case when* $X = \mathrm{CF} - \lambda$*, note that* τ *is nonerasing, since the artificial introduction of the empty-word marker* Λ *is not necessary.*

Finally, we give some simple relations between valence automata and the "corresponding" valence grammars. Let $\mathcal{L}_{left}(\mathrm{Val}, \mathrm{CF}, \mathbf{M})$ be the family of languages generated by leftmost derivations of context-free valence grammars over $\mathbf{M}$.

Theorem 3.7 *1. For any monoid* $\mathbf{M}$, $\mathcal{L}(\mathrm{Val}, \mathrm{REG}, \mathbf{M}) = \mathcal{L}(\mathrm{Val}, \mathrm{NFA}, \mathbf{M})$.

2. For any monoid $\mathbf{M}$, $\mathcal{L}_{left}(\mathrm{Val}, \mathrm{CF}, \mathbf{M}) = \mathcal{L}(\mathrm{Val}, \mathrm{NPDA}, \mathbf{M})$.

Proof. Nearly literally the same construction as in the classical cases (compare with [11, Theorems 9.1,9.2,5.3,5.4]) can be applied, additionally integrating the valences. It has just to be noticed that the construction of a regular grammar from a given DFA [11, Theorem 9.2] works for NFA with λ-moves as well, and that the construction of an NPDA [11, Theorem 5.3] can also be done for arbitrary context-free grammars. □

Corollary 3.8

For any commutative monoid $\mathbf{M}$, $\mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbf{M}) = \mathcal{L}(\mathrm{Val}, \mathrm{NPDA}, \mathbf{M})$.

[1] Possibly, one could call these special valence transductions *valence morphisms*?

Remark 3.9 *Note that the equivalence between pushdown automata and context-free grammars can only be generalized in the case of commutative monoids, when inspecting the classical equivalence proof. One basic reason for this is the fact that in the case of non-commutative monoids, we cannot assume without loss of generality all context-free derivation steps to be leftmost.*

4 Valences over various monoids

4.1 Basic properties

As regards closure properties, valence language classes form semi-AFL's, i.e., they are closed under union and rational transductions. This fact can be shown quite generally, not requiring a separate proof for each monoid, as done in [14, 26].

Theorem 4.1 *For each monoid* $\mathbf{M}$ *and each* $X \in \{\mathrm{REG}, \mathrm{CF}-\lambda, \mathrm{CF}\}$*, the class* $\mathcal{L}(\mathrm{Val}, X, \mathbf{M})$ *is a semi-AFL which is full in the cases* $X = \mathrm{REG}$ *and* $X = \mathrm{CF}$*. Moreover,* $\mathcal{L}(\mathrm{Val}, X, \mathbf{M})$ *is closed under substitution by* $\mathcal{L}(X)$*-languages.*

Analogous results for finite automata with valences can be found in [15].
Proof. (Sketch) The triple constructions known for the basic Chomsky families showing closure under rational transductions given by finite-state transducers in normal form can be adapted for the families of valence languages. Furthermore, the "block coding technique" used for showing closure under restricted erasing can be adapted for our purposes, too.

Thus $\mathcal{L}(\mathrm{Val}, X, \mathbf{M})$ is closed under λ-free transductions for $X = \mathrm{CF} - \lambda$ and under arbitrary rational transductions for $X \in \{\mathrm{REG}, \mathrm{CF}\}$.

As regards the other properties, namely, closure under union and $\mathcal{L}(X)$-substitutions, the standard constructions known from the theory of context-free languages can easily be carried over. □

Since context-free unordered vector languages coincide with context-free valence languages over $\mathbb{Q}_+$, the above reasoning also shows that unordered vector languages form a semi-AFL. This proves that the question marks in the "UV-column" in [3, Table 1] can be replaced by "+", as indicated in the footnote of that page.

There is also a certain simple normal form for context-free grammars with arbitrary valence monoids.

Theorem 4.2 *Let* $\mathbf{M}$ *be an arbitrary monoid. Any language* $L \subseteq \mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbf{M})$ *can be generated by a valence grammar* $G = (N, T, P, S, \mathbf{M})$ *over* $\mathbf{M}$ *with core rules of the forms* $A \to B$*,* $A \to BC$*,* $A \to a$*, and* $A \to \lambda$*, where* $A, B, C \in N$ *and* $a \in T$*.*

Proof. Nearly the same construction as in [11, Theorem 4.5] is applied to transfer rules with right-hand sides of length 2 or greater to rules of the required forms. One way of assigning labels to these new rules would be to give the valence of the original rule to the first of the newly created rules and assign the neutral element of $\mathbf{M}$ to the other rules. $\square$

Note that the proof of the previous theorem depends on the fact that monoids have neutral elements. Hence, it does not carry over to general semigroups. A similar note applies to many proofs of this paper.

Finally, we give two more simple results on the generative power of valences over arbitrary monoids.

Theorem 4.3 *Let* $\mathbf{M}$ *and* $\mathbf{M}'$ *be isomorphic monoids. Then, for* $X \in \{\mathrm{REG}, \mathrm{CF}-\lambda, \mathrm{CF}\}$*,* $\mathcal{L}(\mathrm{Val}, X, \mathbf{M}) = \mathcal{L}(\mathrm{Val}, X, \mathbf{M}')$.

Proof. Let $G = (N, T, P, S, \mathbf{M})$ be a valence grammar over $\mathbf{M}$, and let φ be the isomorphism from $\mathbf{M}$ to $\mathbf{M}'$. In any valence rule of G, replace the valence by its isomorphic image in $\mathbf{M}'$ to obtain $G' = (N, T, P', S, \mathbf{M}')$. It is easily shown by induction on the number of derivation steps that (α, m) is derivable in G iff $(\alpha, \varphi(m))$ is derivable in G'. By interchanging the roles of $\mathbf{M}$ and $\mathbf{M}'$, the other inclusion is shown. $\square$

Theorem 4.4 *Let* $\mathbf{M}$ *be an arbitrary monoid, and let* $\mathcal{F}(\mathbf{M})$ *be the family of finitely generated submonoids of* $\mathbf{M}$*. For* $X \in \{\mathrm{REG}, \mathrm{CF} - \lambda, \mathrm{CF}\}$*,*

$$\mathcal{L}(\mathrm{Val}, X, \mathbf{M}) = \bigcup_{\mathbf{M}' \in \mathcal{F}(\mathbf{M})} \mathcal{L}(\mathrm{Val}, X, \mathbf{M}').$$

Proof. The inclusion $\supseteq$ is trivial. If $L \in \mathcal{L}(\mathrm{Val}, X, \mathbf{M})$, then L is generated by an X-grammar G with valences in $\mathbf{M}$. Obviously, only elements of $\mathbf{M}$ which can be represented as product of rule valences of G can appear in derivations of G. In other words, one could consider the submonoid $\mathbf{M}'$ generated by all the rule valences of G, so that $L \in \mathcal{L}(\mathrm{Val}, X, \mathbf{M}')$, with $\mathbf{M}' \in \mathcal{F}(\mathbf{M})$. $\square$

4.2 Derivation trees

The very useful notion of a derivation tree for a context-free grammar can be generalized as follows. Let $G = (N, T, P, S)$ be a valence grammar over $\mathbf{M}$. A directed tree $D = (V, E)$ is a *derivation tree* for G if:

1. Every node has a *label*, which is a symbol of $N \cup T \cup \{\lambda\}$.
2. Every interior node has a *valence*, which is an element in $\mathbf{M}$.

3. The label of the root is S.
4. If a node is interior and has label A, then A must be in N.
5. If node v has label A and value $\vec{r}$, and if the nodes $v_1, v_2, \ldots, v_k$ are the sons of node v, in order from left to right, with labels $X_1, X_2, \ldots, X_k$, respectively, then $(A \to X_1X_2\ldots X_k, \vec{r})$ must be in P.
6. If node v has label λ, then it is the only son of its father and a leaf.

As in [11, Section 4.3], the leaves of the tree can be ordered from left to right; their labels (in this order) define a word. We also need the concept of a subtree. Let D be a derivation tree and v be a node in D. The subtree of D consisting of root v, all its descendants, the edges connecting them, their labels, and their valences, is denoted by $D(v)$ and referred to as the *subtree of D with root v*. Let v_1 and v_2 be two different nodes in D, where v_2 is a descendent of v_1. The subtree (with labels and valences) induced by v_1 and all its descendants that are not descendants of v_2 is denoted by $D(v_1 - v_2)$ and called the *subtree of D between v_1 and v_2*. A subtree whose root is labelled A is called an *A-tree*.

Next we define *admissible orders* for the nodes of a derivation tree. For a directed tree $D = (V, E)$, the transitive closure E^+ of E is a partial order. A total order on the interior nodes of V is called admissible if it is a refinement of E^+ restricted to the interior nodes of D. An example for an admissible order is the *DFS order*, obtained when traversing the nodes of the tree in depth-first-search (the sons of a node are traversed from left to right). This corresponds to the use of leftmost derivations. A pair $(D, \prec)$ consisting of a derivation tree D and an admissible ordering $\prec$ is called an *ordered* derivation tree. If the interior nodes of D, ordered with respect to $\prec$, are $v_1, v_2 \ldots, v_n$, then the *yield* of $(D, \prec)$ is the pair $(\alpha, m) \in (N \cup T)^* \times M$, where α is the word obtained by reading the labels of the leaves from the left, and $m = m_1 \circ m_2 \ldots \circ m_n$, where m_i is the valence of v_i, $1 \leq i \leq n$.

The yield of a subtree is defined as for a derivation tree. For a derivation tree D in G, we denote by N_D the set of all nonterminals appearing as a label in D. Analogously, for a derivation Δ, let N_Δ be the set of nonterminals appearing in some word during the derivation process.

Theorem 4.5 *Let $G = (N, T, P, S, \mathbf{M})$ be a valence grammar over $\mathbf{M} = (M, \circ, e)$. A derivation of a pair $(\alpha, m) \in (N \cup T)^* \times M$ is possible iff there is an ordered derivation tree with yield (α, m).*

Proof. We prove, more specifically, that a derivation in n steps is possible iff there is a corresponding derivation tree with n interior nodes. As $\mathbf{M}$ is non-commutative, we use a top-down argument, as opposed to the bottom-up strategy in the context-free case [11, Theorem 4.1].

For $n = 0$, the claim is obviously true. Let us suppose that the claim is shown for all $0 \leq n \leq k$, for some specific $k \in \mathbb{N}$.

Consider a $(k+1)$-step derivation $(S, e) \overset{*}{\Rightarrow} (\alpha, m)$. By definition, there are a k-step derivation $(S, e) \overset{*}{\Rightarrow} (\alpha_1 B \alpha_2, m_1)$ and a rule $(B \to \beta, m_2)$ such that $\alpha = \alpha_1 \beta \alpha_2$ and $m = m_1 \circ m_2$. By the induction hypothesis, there is an ordered derivation tree $(D, \prec)$ with n interior nodes and yield $(\alpha_1 B \alpha_2, m_1)$. Let v be the $(|\alpha_1| + 1)$-th leaf of D; its label is B. We give this node a valence of m_2 and add sons labelled β from left to right. The thus created tree D' satisfies the definition of a derivation tree in G. We obtain the order $\prec'$ on the interior nodes of D' by appending v to $\prec$. Obviously, $\prec'$ is admissible. The yield of $(D', \prec')$ is (α, m).

Conversely, in an ordered derivation tree with $(n+1)$ nodes, one can erase those leaves that are sons of the last interior node, and get, by induction, to an equivalent derivation with $n+1$ steps. □

In a similar way, it can be shown:

Theorem 4.6 *Let $G = (N, T, P, S, \mathbf{M})$ be a valence grammar over $\mathbf{M} = (M, \circ, e)$. A leftmost derivation of a pair $(\alpha, m) \in (N \cup T)^* \times M$ is possible iff there is a derivation tree ordered in DFS order with yield (α, m).*

4.3 Valences over finite and commutative monoids

We shall first prove that valence grammars over finite or commutative monoids are not stronger than valence grammars over the corresponding groups. Basically, this is due to the definition of acceptance by the neutral element.[2]

For a monoid $\mathbf{M} = (M, \cdot, e)$, let $E(\mathbf{M})$ be the set of elements that can appear in products yielding e, formally: $E(\mathbf{M}) = \{a \in M : \exists x \exists y (x \cdot a \cdot y = e)\}$. Consider a valence grammar over $\mathbf{M}$. In a derivation with valence e, all applied rules have valences from $E(\mathbf{M})$. We obtain:

Lemma 4.7 *For any monoid $\mathbf{M}$ and $X \in \{\mathrm{CF}, \mathrm{CF} - \lambda, \mathrm{REG}\}$, $\mathcal{L}(\mathrm{Val}, X, \mathbf{M}) = \mathcal{L}(\mathrm{Val}, X, \langle E(\mathbf{M}) \rangle)$.*

Next, we show that $\langle E(\mathbf{M}) \rangle$ is a group if $\mathbf{M}$ is commutative or finite. Hence, valence grammars over finite or commutative monoids are not stronger than valence grammars over finite or commutative groups.

Lemma 4.8 *If $\mathbf{M}$ is commutative, then $E(\mathbf{M})$ is a group.*

[2]In [5, Theorem 5], we claimed that valence grammars over finite monoids and matrix grammars are equivalent. Unfortunately, the idea of the proof is not valid for our acceptance condition. However, it is not very difficult to prove the mentioned equivalence when the acceptance condition is that the valence of a derivation evaluates to a monoid element of a given finite set.

Proof. Consider $a, b \in E(\mathbf{M})$. Choose a_1, a_2, b_1, b_2 such that $a_1 \cdot a \cdot a_2 = b_1 \cdot b \cdot b_2 = 1$. By commutativity, $a_1 \cdot a_2 \cdot b_1 \cdot b_2 \cdot a \cdot b = 1$. Hence, $a \cdot b \in E(\mathbf{M})$ and by $1 \in E(\mathbf{M})$, $E(\mathbf{M})$ is a submonoid of $\mathbf{M}$. It is even a group, as the arbitrarily chosen element a has inverse $a_1 \cdot a_2 \in E(\mathbf{M})$ due to commutativity. □

In the following, let M^M bet the set of functions from M into M. Recall that $(M^M, \circ, \mathrm{id}_M)$ forms a monoid, where $\circ$ is the composition of functions $[f \circ g(x) = g(f(x))]$, and id_M is the identity on M.

Lemma 4.9 *Any monoid* $\mathbf{M} = (M, \cdot, e)$ *is isomorphic to a submonoid of the monoid* $(M^M, \circ, \mathrm{id}_M)$.

Proof. An element $a \in M$ is mapped on $f_a : M \to M$ with $f_a(x) = x \cdot a$, for all $x \in M$. The mapping $a \to f_a$ is a homomorphism, as $f_{a \cdot b} = f_a \circ f_b$. It is injective, as $f_a(e) = a \neq b = f_b(e)$ for $a \neq b$. □

Lemma 4.10 *If* $\mathbf{M} = (M, \circ, e)$ *is finite, then* $E(\mathbf{M})$ *is a finite group.*

Proof. By Lemma 4.9, we can assume that $\mathbf{M}$ is a submonoid of $\mathbf{M}' = (A^A, \circ, \mathrm{id}_A)$, for some finite set A. If $f \in M \subseteq A^A$ is not a permutation (i.e., not surjective) then the range of $f_1 \circ f \circ f_2$ cannot be the whole set A and, thus, $f \notin E(\mathbf{M}) \subseteq E(\mathbf{M}')$. On the other hand, if $f \in M$ is a permutation, then $f^{n_f} = \mathrm{id}_A$, for some $n_f > 0$ and, thus, $f \in E(\mathbf{M})$. Hence, $f \in M$ belongs to $E(\mathbf{M})$ iff f is a permutation. As permutations are closed under composition, $E(\mathbf{M})$ is a submonoid of $\mathbf{M}$. It is also a group, since any $f \in E(\mathbf{M})$ has inverse $f^{n_f - 1}$. □

We could not show whether or not valences over finite *groups* enhance the power of context-free grammars. At least, it is possible to prove a pumping lemma, similar to that of context-free languages.

Theorem 4.11 *Let* $\mathbf{M} = (M, \circ, e)$ *be a finite group. For any language* $L \subseteq \mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbf{M})$*, there is a constant* $n \in \mathbb{N}$ *(depending on* L*) such that:*
For all $z \in L$ *with* $|z| \geq n$*, there is a decomposition* $z = uvwxy$ *with* $|vx| > 0$*,* $|vwx| \leq n$*, and* $uv^iwx^iy \in L$*, for all* $i \geq 1$.

Proof. Let $G = (N, T, P, S, \mathbf{M})$ be a valence grammar over $\mathbf{M}$ which generates L. Without loss of generality (see Theorem 4.2), we can assume that the core rules of G have the forms $A \to BC$, $A \to B$, $A \to a$, or $A \to \lambda$. We choose $n = 2^{|N|(p+1)}$, where $p = |M|$ is the order of the group $\mathbf{M}$.

Consider a word $z \in L$ with $|z| \geq n$ and an ordered derivation tree D of (z, e). As we cannot exclude rules of the forms $A \to B$ and $A \to \lambda$, we must slightly modify the proof of the pumping lemma for context-free languages. The *modified height* $h(t)$ of a node t in D is defined bottom-up as follows:

- For a leaf t, $h(t) = 0$ if t is labelled by λ, and $h(t) = 1$ if t is labelled by $a \in T$.

- If an interior node t has two sons t_1, t_2 with $h(t_1) > 0, h(t_2) > 0$, then $h(t) = \max\{h(t_1), h(t_2)\}+1$. Otherwise, $h(t) = \max\{h(s) : s \text{ is a son of } t\}$.

It is easily shown by induction that the length of the yield of the subtree with root t is bounded by $2^{h(t)-1}$. Hence, for the root r of D, $h(r) \geq |N|(p+1)+1$ must hold. There is a path from r to some leaf such that, for any $i \in \{1, \ldots, h(r)\}$, the path contains a node with modified height i. To construct this path, we start with the root and always choose the son with the greatest modified height. By the pigeonhole principle, the path contains nodes $t_1, \ldots, t_{p+2}$ such that $t_1, \ldots, t_{p+2}$ are labelled by the same symbol $A \in N$ and $|N|(p+1)+1 \geq h(t_1) > h(t_2) > \cdots > h(t_{p+2}) \geq 1$. The subtree between r and t_1 has yield $u'Ay'$, $u', y' \in T^*$, the subtrees between t_i and t_{i+1} $(1 \leq i \leq p+1)$ have the yields v_iAx_i, $v_i, x_i \in T^*$, and the subtree of t_{p+2} yields $w' \in T^*$. We have $|v_1v_2 \cdots v_{p+1}w'x_{p+1} \cdots x_2x_1| \leq n$, as this is the yield of the subtree with root t_1.

Now in G, there are derivations $(A, e) \overset{*}{\Rightarrow} (v_iAx_i, m_i)$ for some $m_i \in \mathbf{M}$ $(1 \leq i \leq p+1)$, and thus also derivations

$$(A, e) \overset{*}{\Rightarrow} (v_i \cdots v_jAx_j \cdots x_i, m_i \circ \cdots \circ m_j), \quad 1 \leq i < j \leq p+1.$$

Again by the pigeonhole principle, there are indices $1 < i \leq j \leq p+1$ such that

$$m_1 \circ \cdots \circ m_{i-1} = m_1 \circ \cdots \circ m_j =: m.$$

Hence, $m(m_i \circ \cdots \circ m_j) = m$. Since $\mathbf{M}$ is a group, this implies $m_i \circ \cdots \circ m_j = e$ and, moreover, the existence of a derivation $(A, e) \overset{*}{\Rightarrow} (v_i \cdots v_jAx_j \cdots x_i, e)$ in G. The desired decomposition $z = uvwxy$ is now found as $u = u'v_1 \cdots v_{i-1}$, $v = v_i \cdots v_j$, $w = v_{j+1} \cdots v_{p+1}w'x_{p+1} \cdots x_{j+1}$, $x = x_j \cdots x_i$, $y = x_{i-1} \cdots x_1y'$. □

Remark 4.12 *Note that the construction does not imply $uwy \in L$, as the derivation $(A, e) \overset{*}{\Rightarrow} (vAx, e)$ is not a subderivation of the original derivation.*

Finally, we are going to show that the family of languages generated by valence grammars of type $X \in \{\mathrm{REG}, \mathrm{CF} - \lambda, \mathrm{CF}\}$ over some commutative monoid is either $\mathcal{L}(\mathrm{Val}, X, \mathbb{Q}_+)$ or $\mathcal{L}(\mathrm{Val}, X, \mathbb{Z}^k)$, for some $k \geq 0$. This implies that our results on valence grammars over $\mathbb{Z}^k$ as presented in the following two sections are of a quite general nature. Firstly, we shall prove that derivations in valence grammars over commutative monoids can be analyzed in a bottom-up manner. As the operation in a commutative monoid is independent of the operands' order, we obtain as a special case of Theorem 4.5:

Corollary 4.13 *Let $G = (N, T, P, S, \mathbf{M})$ be a valence grammar over a commutative monoid $\mathbf{M} = (M, \circ, e)$. A derivation of a pair $(\alpha, m) \in (N \cup T)^* \times M$ is possible iff there is an ordered derivation tree with nodes in DFS order yielding (α, m).*

Theorem 4.14 *Let $G = (N, T, P, S, \mathbf{M})$ be a valence grammar over a commutative monoid $\mathbf{M} = (M, \circ, e)$. A derivation $(A, e) \overset{*}{\Rightarrow} (\alpha, m)$ exists iff there are a rule $(A \to X_1 \cdots X_k, m_0)$, $X_1, \ldots, X_k \in N \cup T$, and derivations*

$$(X_1, e) \overset{*}{\Rightarrow} (\alpha_1, m_1), \ldots, (X_k, e) \overset{*}{\Rightarrow} (\alpha_k, m_k)$$

such that $\alpha = \alpha_1 \cdots \alpha_k$ and $m = m_0 \circ m_1 \circ \cdots \circ m_k$.

As a first application of Theorem 4.14, we show the following result:

Lemma 4.15 *Let $\mathbf{M} = (M, \circ, e)$ be a finite commutative monoid, and let $k \in \mathbb{N}$. Then, $\mathcal{L}(\mathrm{Val}, X, \mathbf{M} \times \mathbb{Z}^k) = \mathcal{L}(\mathrm{Val}, X, \mathbb{Z}^k)$ for $X \in \{\mathrm{CF} - \lambda, \mathrm{CF}\}$.*

Proof. The inclusion $\mathcal{L}(\mathrm{Val}, X, \mathbb{Z}^k) \subseteq \mathcal{L}(\mathrm{Val}, X, \mathbf{M} \times \mathbb{Z}^k)$ is trivial. Let $G = (N, T, P, S, \mathbf{M} \times \mathbb{Z}^k)$ be a valence grammar over $\mathbf{M} \times \mathbb{Z}^k$, with core rules of the forms $A \to BC$, $A \to B$, $A \to a$, $A \to \lambda$, see Theorem 4.2. We construct the valence grammar $G' = (N', T, P', S', \mathbb{Z}^k)$ as follows:

- $N' = N \times M$,
- For any rule $(A \to BC, (m, \vec{v})) \in P$, P' contains all rules $((A, m_0) \to (B, m_1)(C, m_2), \vec{v})$ with $m_0 = m \circ m_1 \circ m_2$.
- For any rule $(A \to B, (m, \vec{v})) \in P$, P' contains all rules $((A, m_0) \to (B, m_1), \vec{v})$ with $m_0 = m \circ m_1$.
- For any rule $(A \to a, (m, \vec{v})) \in P$, P' contains the rule $((A, m) \to a, \vec{v})$.
- For any rule $(A \to \lambda, (m, \vec{v})) \in P$, P' contains the rule $((A, m) \to \lambda, \vec{v})$.
- $S' = (S, e)$.

By a bottom-up induction as indicated in Theorem 4.14, it can be shown that $((A, m), \vec{0}) \overset{*}{\Rightarrow} (w, \vec{v})$ holds in G' for $w \in T^*$, $A \in N$, $m \in M$, $\vec{v} \in \mathbb{Z}^k$, iff $(A, \vec{0}) \overset{*}{\Rightarrow} (w, (m, \vec{v}))$ holds in G. □

Before proving the main result of this section, we give some auxiliary results from the theory of commutative monoids. The first lemma is known as the Fundamental Theorem for finitely generated Abelian (i.e., commutative) groups [20].

Lemma 4.16 *Any finitely generated commutative group is isomorphic to some group* $\mathbf{M} \times \mathbb{Z}^k$*, where* $k \geq 0$ *and* $\mathbf{M}$ *is a finite commutative group.*

Theorem 4.17 *Let* $\mathbf{M}$ *be a commutative monoid and* $X \in \{\mathrm{REG}, \mathrm{CF} - \lambda, \mathrm{CF}\}$*. Then, the class* $\mathcal{L}(\mathrm{Val}, X, \mathbf{M})$ *equals either* $\mathcal{L}(\mathrm{Val}, X, \mathbb{Q}_+)$ *or* $\mathcal{L}(\mathrm{Val}, X, \mathbb{Z}^k)$ *for some* $k \geq 0$.

Proof. By Lemma 4.8, without loss of generality, we can assume that $\mathbf{M}$ is a group. Any valence grammar G over $\mathbf{M}$ is a valence grammar over the finitely generated commutative group $\langle \mathrm{Lab}(G) \rangle$, see Theorem 4.3. According to Lemma 4.16, $\langle \mathrm{Lab}(G') \rangle$ is isomorphic to some $\mathbf{M} \times \mathbb{Z}^k$, where $\mathbf{M}$ is a finite commutative group, and $k \geq 0$. Finally, by Lemma 4.15, there is an equivalent valence grammar over $\mathbb{Z}^k$.

Now, there are two possibilities for the finitely generated subgroups of $\mathbf{M}$. Possibly, there is a $k \in \mathbb{N}$ such that

$$(*) \text{ any of these groups is isomorphic to some } \mathbf{N} \times \mathbb{Z}^i,$$

where $\mathbf{N}$ is a finite commutative group and $i \leq k$. Choose the smallest $k \in \mathbb{N}$ such that $(*)$ holds. Then, $\mathcal{L}(\mathrm{Val}, X, \mathbf{M}) = \mathcal{L}(\mathrm{Val}, X, \mathbb{Z}^k)$. Otherwise, $\mathcal{L}(\mathrm{Val}, X, \mathbf{M}) = \mathcal{L}(\mathrm{Val}, X, \mathbb{Q}_+)$. □

We close this section by stating some remarkable closure properties of families of valence languages over $\mathbb{Q}_+$.

Theorem 4.18 *The families* $\mathcal{L}(\mathrm{Val}, X, \mathbb{Q}_+)$*,* $X \in \{\mathrm{REG}, \mathrm{CF}\}$*, are closed under valence transductions over* $\mathbb{Q}_+$*.* $\mathcal{L}(\mathrm{Val}, \mathrm{CF} - \lambda, \mathbb{Q}_+)$ *is closed under non-erasing valence transductions over* $\mathbb{Q}_+$*.*

Proof. Firstly, note that, for a valence grammar (transducer) over $\mathbb{Q}_+$, an equivalent valence grammar (transducer) exists over some $\mathbb{Z}^k$ for some $k \geq 0$ and vice versa. Let G be a valence grammar over $(\mathbb{Z}^k, +, \vec{0})$ and $\mathcal{A}$ be a valence transducer over $(\mathbb{Z}^l, +, \vec{0})$. Again, the triple construction for the classical language families can be modified to obtain a valence grammar H over $(\mathbb{Z}^{k+l}, +, \vec{0})$ such that $L(H) = \tau_{\mathcal{A}}(L(G))$. Note that the commutativity of addition is essential for the construction. □

Remark 4.19 *By analyzing the proof of the preceding theorem, one could state even a bit more generally:*

If $L \in \mathcal{L}(\mathrm{Val}, X, \mathbf{M})$*,* $X \in \{\mathrm{REG}, \mathrm{CF}\}$*, where* $\mathbf{M}$ *is an arbitrary monoid, and* $\mathbf{M}'$ *is a commutative monoid such that* τ *is a valence transduction over* $\mathbf{M}'$*, then*

$$\tau(L) \in \mathcal{L}(\mathrm{Val}, X, \mathbf{M} \times \mathbf{M}').$$

Corollary 4.20 *Each of the families $\mathcal{L}(\mathrm{Val}, X, \mathbb{Q}_+)$, $X \in \{\mathrm{REG}, \mathrm{CF}\}$, is closed under permutation and under intersection with languages from $\mathcal{L}(\mathrm{Val}, \mathrm{REG}, \mathbb{Q}_+)$.*

This generalizes the older result that *BLIND* is closed under intersection [9].
Proof. In Section 3, it was stated that the mentioned operations are valence transductions. Therefore, the previous theorem yields the claim. □

5 Normal forms for valence grammars over $\mathbb{Z}^k$

5.1 Statement of the main results

In this section, we shall construct Chomsky and Greibach normal forms for valence grammars over $\mathbb{Z}^k$. More specifically, we prove:

Theorem 5.1 *For any valence grammar over $\mathbb{Z}^k$, there are:*

1. *(Chomsky NF I) an equivalent valence grammar over $\mathbb{Z}^k$ with valence rules of the forms $(A \to BC, \vec{r})$, $\|\vec{r}\|_1 \leq 1$ or $(A \to a, \vec{0})$,*
2. *(Chomsky NF II) an equivalent valence grammar over $\mathbb{Z}^k$ with valence rules of the forms $(A \to BC, \vec{0})$ or $(A \to a, \vec{r})$, $\|\vec{r}\|_1 \leq 1$, as well as*
3. *(Greibach NF) an equivalent valence grammar over $\mathbb{Z}^k$ with valence rules of the forms $(A \to a\alpha, \vec{r})$, $\|\vec{r}\|_1 \leq 1$.*

In the preceding theorem and in what follows, we use the convention that $A, B, C, \ldots$ denote nonterminal symbols, $a, b, c, \ldots$ denote terminal symbols, α, β, γ denote words consisting of nonterminal symbols, and $u, v, x, \ldots$ denote (possibly empty) words consisting of terminal symbols.

In [2], it is shown that, for any valence grammar with unit or zero valence vectors, there is an equivalent unordered vector grammar with the same set of core rules. This implies:

Corollary 5.2 *For any unordered vector grammar, there are equivalent unordered vector grammars*

1. *in Chomsky normal form, i.e., with core rules of the form $A \to BC$ or $A \to a$, as well as*
2. *in Greibach normal form, i.e., with core rules of the form $A \to a\alpha$.*

Remark 5.3 *The Greibach normal form immediately yields a simple machine characterization of valence languages over $\mathbb{Z}^k$ via pushdown machines (without λ-steps, i.e., working in real-time) endowed with k blind counters. The proofs of [11, Theorems 5.3,5.4] can easily be extended to valence grammars and pushdown automata.*

As regards language family hierarchies, we obtain via [24, 25]:

Corollary 5.4 *Let $k \geq 0$. Then, we have:*

$$\begin{array}{rcl} \mathcal{L}(\mathrm{Val}, \mathrm{CF}-\lambda, \mathbb{Z}^k) & = & \mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbb{Z}^k) \quad \subset \quad \mathcal{L}(\mathrm{Val}, \mathrm{CF}-\lambda, \mathbb{Z}^{k+1}) \\ & = & \mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbb{Z}^{k+1}) \quad \subset \quad \mathcal{L}(\mathrm{Val}, \mathrm{CF}-\lambda, \mathbb{Q}_+) \\ & = & \mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbb{Q}_+) \quad = \quad \mathcal{L}(\mathrm{UV}, \mathrm{CF}-\lambda) \\ & = & \mathcal{L}(\mathrm{UV}, \mathrm{CF}) \quad \subset \quad \mathrm{LOG}(\mathrm{CFL}). \end{array}$$

To keep the construction proving the stated main results readable, we have split it into three phases, organized in the according subsections. In the first phase of the construction of the normal form, the erasing rules are replaced; then, unit productions are eliminated; lastly, the valences are normalized.

5.2 Discussion of the main results

Computational complexity. Besides being an interesting result in itself, the normal form theorem also has consequences regarding computational complexity. Sudborough [24] showed that, in analogy to the context-free case, the membership problem can be efficiently solved for unordered vector grammars without erasing and unit productions or, more exactly, it was shown to be in LOG(CFL). Satta [23] could prove that complexity result for arbitrary unordered vector grammars, while leaving open the problem of the existence of normal forms. Our result yields an alternative proof for the relation $\mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbb{Q}_+) \subseteq \mathrm{LOG}(\mathrm{CFL})$.

Parsing of valence languages. Another idea would be to develop parsing algorithms for context-free valence grammars in Chomsky normal form, as had been done by Cocker, Younger and Kasami for the context-free case, yielding the so-called CYK procedure, which is basically a dynamic programming algorithm whose main data structure is usually called CYK table, see [11]. A straightforward adaptation of their dynamic programming method gives an algorithm requiring $O(n^{2k+3})$ time and $O(n^{k+2})$ space:

Namely, for each subword u of a given word w, one has to store the $O(n^k)$ pairs $(A, \vec{r})$ with $(A, \vec{0}) \overset{*}{\Rightarrow} (u, \vec{r})$. To compute the entry at position (i, j) $(1 \leq i < j \leq n)$ of the CYK table, one needs to study all pairs $((B, \vec{r}), (C, \vec{s}))$, where $(B, \vec{r})$ appears at position (i, ℓ), and $(C, \vec{s})$ at position $(\ell + 1, j)$, for $i \leq \ell \leq j$. The total time to compute the entry at a specific position thus amounts to $O(n^{2k} \cdot n)$, which yields $O(n^{2k+3})$ time for the complete algorithm.

An open question: can the normal form theorem be generalized? The proof of the normal form theorem often uses bottom-up induction as indicated in Theorem 4.14. Thus, it cannot be modified for valence grammars over non-commutative monoids. This remains a challenging open question. Closely related are questions concerning Chomsky or Greibach normal forms for matrix or vector grammars, see [2].

5.3 Elimination of erasing rules

The first proposition shows that, given a valence grammar over $\mathbb{Z}^k$, an equivalent valence grammar over $\mathbb{Z}^k$ can be found whose set of nonterminals is partitioned into a set generating only non-empty terminal words, and a second set generating the empty word λ as the only terminal word.

Proposition 5.5 *Let G be a context-free valence grammar over $\mathbb{Z}^k$. Then, there is a valence grammar $H = (N, T, P, S)$ over $\mathbb{Z}^k$ with $L(H) = L(G) \setminus \{\lambda\}$ such that*

- $N = N_1 \cup N_2,\ N_1 \cap N_2 = \emptyset$
- *The core rules of P have one of the following forms:*
  - $A \to BC,\ A, B, C \in N, A \in N_1 \wedge (B \in N_1 \vee C \in N_1)$
  - $A \to BC,\ A \in N_2 \wedge B \in N_2 \wedge C \in N_2$
  - $A \to B,\ A \in N_1 \wedge B \in N_1$
  - $A \to B,\ A \in N_2 \wedge B \in N_2$
  - $A \to a,\ A \in N_1 \wedge a \in T$
  - $A \to \lambda,\ A \in N_2$
- *The axiom S of H lies in N_1.*

Proof. Let $G = (N_1, T, P_1, S_1, \mathbb{Z}^k)$, with core rules as in Theorem 4.2. We choose $H = (N, T, P, S, \mathbb{Z}^k)$ as follows. $N = N_1 \cup N_2$ where N_2 is a disjoint copy of N_1, the copy of $A \in N_1$ is denoted by A', $S = S_1$, and P is defined as:

$$\begin{aligned}
(A \to BC, \vec{r}) \in P_1 &\iff (A \to BC, \vec{r}), (A \to B'C, \vec{r}),\\
&\qquad (A \to BC', \vec{r}), (A' \to B'C', \vec{r}) \in P,\\
(A \to B, \vec{r}) \in P_1 &\iff (A \to B, \vec{r}), (A' \to B', \vec{r}) \in P,\\
(A \to a, \vec{r}) \in P_1 &\iff (A \to a, \vec{r}) \in P,\\
(A \to \lambda, \vec{r}) \in P_1 &\iff (A' \to \lambda, \vec{r}) \in P.
\end{aligned}$$

It can be easily shown by bottom-up induction on the number of derivation steps that a derivation $(A, \vec{0}) \overset{*}{\Rightarrow}_H (w, \vec{r})$, $A \in N_1, w \in T^*$ is possible iff $w \neq \lambda$ and there is a derivation $(A, \vec{0}) \overset{*}{\Rightarrow}_G (w, \vec{r})$, while a derivation $(A', \vec{0}) \overset{*}{\Rightarrow}_H (w, \vec{r})$, $A' \in N_2, w \in T^*$ exists iff $w = \lambda$ and there is a derivation $(A, \vec{0}) \overset{*}{\Rightarrow}_G (\lambda, \vec{r})$. □

Let us consider the tree of an arbitrary derivation in the grammar H of the previous proposition. Let v be a node labeled by $A \in N_1$ having two sons, v_1 and v_2, labeled by $B' \in N_2$ and $C \in N_1$, respectively. The idea of the construction of the λ-free grammar is to delete in the derivation tree the subtree with the root v_1 and to insert a path from v to v_2 whose valence is that of the deleted subtree. For this purpose, the following proposition is of crucial importance.

Proposition 5.6 *Let $G = (N, T, P, S, \mathbb{Z}^k)$ be a context-free valence grammar with core rules of the forms $A \to BC$, $A \to B$, and $A \to \lambda$. There is a regular valence grammar $G' = (N', T', P', S', \mathbb{Z}^k)$ with core rules of the forms $A \to B$ and $A \to \lambda$ such that $(S, \vec{0}) \overset{*}{\Rightarrow}_G (\lambda, \vec{r})$ iff $(S', \vec{0}) \overset{*}{\Rightarrow}_{G'} (\lambda, \vec{r})$. Moreover, if $(\lambda, \vec{r})$ can be derived in G, it can be obtained in G' in* $\max\{1, ||\vec{r}||_1\}$ *steps.*

Proof. Let $U = \{a_1, \ldots, a_k\} \cup \{b_1, \ldots, b_k\}$. We define the mappings word : $\mathbb{Z}^k \to U^*$ and vec : $U^* \to \mathbb{Z}^k$ by

$$\begin{aligned} \text{word}(\vec{r}) &= c_1^{|r_1|} \ldots c_k^{|r_k|}, \text{ where } \left\{ \begin{array}{lcl} c_i &=& a_i \text{ if } r_i \geq 0 \text{ and} \\ c_i &=& b_i \text{ if } r_i < 0, \end{array} \right. \quad \text{and } \vec{r} = (r_1, \ldots, r_k), \\ \text{vec}(w) &= (|w|_{a_1} - |w|_{b_1}, \ldots, |w|_{a_k} - |w|_{b_k}). \end{aligned}$$

Note that the mapping $\tau : U^* \to U^*$ with $\tau(w) = \text{word}(\text{vec}(w))$ is a rational valence transduction over $\mathbb{Z}^k$.

In the first step, we construct the context-free grammar $G_1 = (N, U, P_1, S)$, with $P_1 = \{A \to \text{word}(\vec{r})\alpha : (A \to \alpha, \vec{r}) \in P\}$. Then, we can find a right-linear grammar $G_2 = (N', U, P_2, S')$ generating a symbol equivalent language [22].

Finally, we can define the regular valence grammar $G' = (N', T, P', S', \mathbb{Z}^k)$ with $L(G') = \tau(L(G_2))$ according to Theorem 3.5. A pair $(\lambda, \vec{r})$ can be derived in G' iff there is some $w \in L(G_2)$ with $\text{vec}(w) = \vec{r}$, i.e., by the symbol equivalence of $L(G_1)$ and $L(G_2)$, iff $\text{word}(\vec{r}) \in L(G_1)$. Moreover, note that $|\text{word}(\vec{r})| = ||\vec{r}||_1$ and that, without loss of generality, any word $w \in L(G_2)$ is generated in $\max\{|w|, 1\}$ steps. We can conclude that $(S, \vec{0}) \overset{*}{\Rightarrow} (\lambda, \vec{r})$ in G iff $(S', \vec{0}) \overset{*}{\Rightarrow} (\lambda, \vec{r})$ in G' (in $\max\{||\vec{r}||_1, 1\}$ steps). □

Proposition 5.7 *For any context-free valence grammar G over $\mathbb{Z}^k$, there is a valence grammar G' over $\mathbb{Z}^k$ with core rules of the forms $A \to BC$ or $A \to B$ or $A \to a$ such that $L(G') = L(G) \setminus \{\lambda\}$.*

Proof. Without loss of generality, we can assume that $G = (N, T, P, S, \mathbb{Z}^k)$ has the form as in Proposition 5.5. For any $B' \in N_2$, we define the context-free valence grammar $G_{B'} = (N_2, T, P_2, B', \mathbb{Z}^k)$, where P_2 is the set of all rules whose left-hand side is an element of N_2. By Proposition 5.6, there is a regular valence grammar $G'_{B'} = (N_{B'}, T, P_{B'}, S_{B'})$ such that $(B', \vec{0}) \overset{*}{\Rightarrow} (\lambda, \vec{r})$ holds in $G_{B'}$ iff $(S_{B'}, \vec{0}) \overset{*}{\Rightarrow} (\lambda, \vec{r})$ holds in $G'_{B'}$. Without loss of generality, we assume that $N_{B'}$ and $N_{C'}$ are disjoint for different $B', C' \in N_2$, and set $M = \bigcup_{B' \in N_2} N_{B'}$, $Q = \bigcup_{B' \in N_2} P_{B'}$.

Now we are ready to construct the λ-free valence grammar $G' = (N', T, P', S', \mathbb{Z}^k)$. We choose the set of nonterminals as $N' = N_1 \cup M \times N_1$. P' consists of P_1, the set of all rules in P with a left-hand side from N_1, and of the following rules:

$$\begin{aligned}
(A \to (S_{B'}, C), \vec{r}) \in P' &\iff (A \to B'C, \vec{r}) \in P \text{ or } (A \to CB', \vec{r}) \in P,\\
((X, C) \to (Y, C), \vec{r}) \in P' &\iff (X \to Y, \vec{r}) \in Q,\\
((X, C) \to C, \vec{r}) \in P' &\iff (X \to \lambda, \vec{r}) \in Q.
\end{aligned}$$

G and G' are equivalent, since any leftmost derivation $(A, \vec{0}) \Rightarrow (B'C, \vec{r}) \Rightarrow^* (C, \vec{r}+\vec{s})$ in G can be replaced by the leftmost derivation $(A, \vec{0}) \Rightarrow ((S_{B'}, C), \vec{r}) \Rightarrow^* (C, \vec{r} + \vec{s})$ in G', and vice versa. □

5.4 Elimination of unit productions

In this subsection, we are going to substitute the unit productions of the form $(A \to B, \vec{r})$, where A and B are nonterminals. The basic idea is to construct, for a given valence grammar $G = (N, T, P, S)$ over $\mathbb{Z}^k$, a context-free valence grammar $G_{M,2}$ in Chomsky normal form and a regular valence grammar $G_{M,1}$, for all $M \subseteq N$. A pair $(w, \vec{0})$ can be derived in G iff there are $M \subseteq N$ and $\vec{r} \in \mathbb{Z}^k$ such that $(w, \vec{r})$ is derivable in $G_{M,2}$ and $(\lambda, -\vec{r})$ is derivable in $G_{M,1}$. From $G_{M,1}$ and $G_{M,2}$ we construct a valence grammar G_M in Chomsky normal form generating $(w, \vec{0})$ iff $(\lambda, -\vec{r})$ and $(w, \vec{r})$, respectively, are derivable in $G_{M,1}$, $G_{M,2}$, respectively, for some $\vec{r}$. Finally, a valence grammar G' in normal form is constructed generating $\bigcup_{M \subseteq N} L(G_M) = L(G)$.

We start with some useful definitions. Let $G = (N, T, P, S, \mathbb{Z}^k)$ be a valence grammar over $\mathbb{Z}^k$ with rules of the form as in Proposition 5.7. Let D be a derivation tree in G. If a subtree $D(v_1 - v_2)$ is a nontrivial path and v_1 and v_2 have the same label, it is called a *loop-path*. A derivation is called *loop-free* if the corresponding derivation tree does not contain any loop-path. For a nonterminal

$A \in N$ and for a subset $M \subseteq N$, we define

$$\begin{aligned} LOOP(A) &= \{ \vec{r} \in \mathbb{Z}^k : (A, \vec{0}) \overset{*}{\Rightarrow} (A, \vec{r}) \}, \\ LOOP(M) &= \{ \vec{r} \in \mathbb{Z}^k : \exists A_1, \ldots, A_n \in M \exists \vec{r_1} \in LOOP(A_1), \ldots, \vec{r_n} \in LOOP(A_n) \\ &\quad \vec{r} = \sum_{i=1}^{n} \vec{r_i} \}. \end{aligned}$$

Proposition 5.8 *Let $G = (N, T, P, S)$ be a λ-free valence grammar over $\mathbb{Z}^k$. Then $(S, \vec{0}) \overset{*}{\Rightarrow} (w, \vec{0})$, $w \in T^+$, iff there is a loop-free derivation Δ yielding $(w, \vec{r})$, where $-\vec{r} \in LOOP(N_\Delta)$ and N_Δ is the set of nonterminals occurring in Δ.*

Proof. Let $\vec{r} \in LOOP(M)$, with $\vec{r} = \sum_{i=1}^{n} \vec{r_i}$, $\vec{r_i} \in LOOP(A_i)$, $A_i \in M$ and $i \leq n$, for some $M \subseteq N$. Let D be a derivation tree with $M \subseteq N_D$ (where N_D is the set of nonterminals in D) yielding $(w, \vec{s})$. By induction over n, it is easily shown that there exists a derivation tree D' yielding $(w, \vec{s} + \vec{r})$.

On the other hand, let D_0 be a derivation tree yielding $(w, \vec{0})$. We shall construct a (finite) sequence $D_0, D_1, \ldots, D_t$ of derivation trees such that

1. The yield of D_i is $(w, \vec{s_i})$, with $\vec{s_i} \in LOOP(N_{D_t})$, for all $1 \leq i \leq t$,
2. D_t has no loop-paths.

If the tree D_i contains a loop-path, we define D_{i+1} as follows. Let V_i be the set of all nodes v in D_i that are starting points of some loop-path.

Let v_i be the minimal node (in DFS order) which is a starting point of some loop-path, and let u_i be the maximal end point of a loop-path starting at v_i. Let $\vec{r_i}$ be the valence of this path, and let A_i be the common label of v_i and u_i. D_{i+1} is obtained by cutting the path $D_i(v_i - u_i)$ from D_i, i.e., the subtree with root v_i is removed and replaced by the subtree with root u_i.

The newly obtained tree D_{i+1} is obviously a derivation tree in G yielding $(w, \vec{s}_{i+1})$ with $\vec{s}_{i+1} = \vec{s_i} - \vec{r_i}$. Since D_{i+1} has less nodes than D_i, there must be some t such that D_t has no loop paths.

Note that the choice of v_i and u_i guarantees that the node u_i is not deleted in the further process. Hence, the node set of D_t contains u_i, for all $0 \leq i \leq t-1$. Therefore, $A_i \in N_{D_t}$, for all $0 \leq i \leq t-1$. It follows that $\mathrm{val}(D_t) = -\sum_{i=0}^{t-1} \vec{r_i}$. Since the vectors $\vec{r_i}$ belong to $LOOP(A_i)$ and $A_i \in N_{D_t}$, $1 \leq i \leq s$, we get $-\mathrm{val}(D_t) \in LOOP(N_{D_t})$. □

Proposition 5.9 *Let G be a valence grammar over $\mathbb{Z}^k$ with core rules of the forms $A \to BC$, $A \to B$, $A \to a$. For any $M \subseteq N$, there is a regular valence grammar $G_{M,1}$ over $\mathbb{Z}^k$ such that a pair $(\lambda, \vec{r})$ is derived in $G_{M,1}$ iff $\vec{r} \in LOOP(M)$; in this case, the number of derivation steps is* $\max\{1, ||\vec{r}||_1\}$.

Proof. Let $G = (N, T, P, S, \mathbb{Z}^k)$. For $A \in N$, set $G_A = (N, T, P_A, A, \mathbb{Z}^k)$ with $P_A = \{(B \to C, \vec{r}) \in P\} \cup \{(A \to \lambda, \vec{0})\}$. A pair $(\lambda, \vec{r})$ is generated in G_A iff $\vec{r} \in \mathit{LOOP}(A)$. By renaming the symbols in G_A, we obtain valence grammars $G_A = (N_A, T, P'_A, S_A, \mathbb{Z}^k)$, such that $N_A \cap N_B = \emptyset$ for $A \neq B$. For $M \subseteq N$, let G_M be the valence grammar with the set of nonterminals $\bigcup_{A \in M} N_A \cup \{S\}$, the set of valence rules $\bigcup_{A \in M} P_A \cup \{(S \to S_A S, \vec{0}), (S \to S_A, \vec{0}) : A \in M\}$, and the start symbol S. Obviously, $(\lambda, \vec{r})$ is generated in G_M iff $\vec{r} \in \mathit{LOOP}(M)$.

The regular valence grammar $G_{M,1}$ with the properties mentioned above can be constructed as in the proof of Proposition 5.6. □

Proposition 5.10 *Let G be a valence grammar over $\mathbb{Z}^k$ with core rules of the forms $A \to BC$, $A \to B$, $A \to a$. For any $M \subseteq N$, there is a valence grammar $G_{M,2}$ over $\mathbb{Z}^k$ such that:*

1. *For any pair $(w, \vec{r})$, there is a derivation in $G_{M,2}$ iff there is a loop-free derivation Δ in G for this pair with $M \subseteq N_\Delta$.*
2. *The core rules of $P_{M,2}$ have the forms $A \to BC$ or $A \to a$.*

Proof. Let $G = (N, T, P, S, \mathbb{Z}^k)$. Consider the following set

$$Q \subseteq N \times (N \cup T)^* \times \mathbb{Z}^k \times \mathcal{P}(N).$$

For any loop-free derivation $\Delta : (A, \vec{0}) \stackrel{*}{\Rightarrow} (A', \vec{r'}) \Rightarrow (\alpha, \vec{r})$ with $\alpha \in T \cup N^2$, Q contains $(A \to \alpha, \vec{r}, N_\Delta)$.

Set $G_{M,2} = (N \times \mathcal{P}(N), T, P', (S, M))$, where P' contains all valence rules

- $((A, M_0) \to (B, M_1)(C, M_2), \vec{r})$ with $(A \to BC, \vec{r}, M_3) \in Q$ and $M_0 \subseteq M_1 \cup M_2 \cup M_3$ for some $M_3 \subseteq N$, and
- $((A, M_0) \to a, \vec{r})$ with $(A \to a, \vec{r}, M_1) \in Q$ and $M_0 \subseteq M_1$ for some $M_1 \subseteq N$.

By bottom-up induction on the number of derivation steps, it can be shown that $((A, M_0), \vec{0}) \stackrel{*}{\Rightarrow} (w, \vec{r})$ in $G_{M,2}$ iff there is a loop-free derivation $\Delta : (A, \vec{0}) \stackrel{*}{\Rightarrow} (w, \vec{r})$ in G with $M_0 \subseteq N_\Delta$. □

Proposition 5.11 *Let c be a positive integer, let $G_1 = (N_1, T, P_1, S_1, \mathbb{Z}^k)$ be a regular valence grammar over $\mathbb{Z}^k$, and let $G_2 = (N_2, T, P_2, S_2, \mathbb{Z}^k)$ be a context-free valence grammar over $\mathbb{Z}^k$ with core rules of the forms $A \to BC$, $A \to a$.*

Then, there is a context-free valence grammar $G = (N, T, P, S, \mathbb{Z}^k)$ with core rules of the forms $A \to BC$, $A \to a$, generating a pair $(w, \vec{r})$, $w \in T^*$, $\vec{r} \in \mathbb{Z}^k$, iff there are $\vec{r}_1, \vec{r}_2$ such that (1) $(\lambda, \vec{r}_1)$ is derivable in G_1 in at most $c|w|$ steps, (2) $(w, \vec{r}_2)$ is derivable in G_2 and (3) $\vec{r} = \vec{r}_1 + \vec{r}_2$.

Proof. Given G_1 and G_2, the desired grammar G is obtained by the following triple construction:

$$\begin{aligned}
N &= (N_1 \cup \{\lambda\}) \times N_2 \times (N_1 \cup \{\lambda\}),\\
S &= (S_1, S_2, \lambda),\\
P &= \big\{((A_1, A, A_2) \to (A_1, B, B_2)(B_2, C, A_2), \vec{r}) :\\
&\qquad A_1, A_2, B_2 \in N_1 \cup \{\lambda\}, A, B, C \in N_2, (A \to BC, \vec{r}) \in P_2\big\}\\
&\cup \ \big\{((A_1, A, A_2) \to a, \vec{r}) : A_1, A_2 \in N_1 \cup \{\lambda\}, A \in N_2, a \in T,\\
&\qquad\qquad (A_1, \vec{0}) \overset{*}{\Rightarrow}_{G_1} (A_2, \vec{r}_1) \text{ in maximal } c \text{ steps},\\
&\qquad\qquad (A \to a, \vec{r}_2) \in P_2, \vec{r} = \vec{r}_1 + \vec{r}_2\big\}.
\end{aligned}$$

Again, by bottom-up induction, it is provable that $((A_1, A, A_2), \vec{0}) \overset{*}{\Rightarrow}_G (w, \vec{r})$ iff there are $\vec{r}_1, \vec{r}_2$ such that (1) $(A_1, \vec{0}) \overset{*}{\Rightarrow}_{G_1} (A_2, \vec{r}_1)$ in at most $c|w|$ steps, (2) $(A, \vec{0}) \overset{*}{\Rightarrow}_{G_2} (w, \vec{r}_1)$ and (3) $\vec{r} = \vec{r}_1 + \vec{r}_2$. □

Proposition 5.12 (Chomsky normal form) *For any context-free valence grammar G over $\mathbb{Z}^k$, there is an equivalent context-free valence grammar G over $\mathbb{Z}^k$ with core rules of the form $A \to BC$, $A \to a$.*

Proof. Let $G = (N, T, P, S, \mathbb{Z}^k)$. For any subset M of N, let $G_{M,1}$ and $G_{M,2}$ be constructed as in Propositions 5.9 and 5.10. Then, for $w \in T^*$, there is a derivation Δ of $(w, \vec{0})$ in G with $M \subseteq N_\Delta$ iff there is an $\vec{r} \in \mathbb{Z}^k$ such that (1) $(\lambda, \vec{r})$ is generated in $G_{M,1}$ and (2) $(w, -\vec{r})$ is generatable in $G_{M,2}$. Since $(w, -\vec{r})$ is generated in $(2|w| - 1)$ steps in $G_{M,2}$, the inequality $||\vec{r}||_1 \leq c'(2|w| - 1)$ holds, where c' is the greatest 1-norm appearing in the rules of $G_{M,2}$. Hence, $(\lambda, \vec{r})$ is generated in $G_{M,1}$ in less than $2c'|w|$ steps.

With $c = 2c'$, according to Propositions 5.9, 5.10 and 5.11, we can construct a valence grammar $G_M = (N_M, T, P_M, S_M, \mathbb{Z}^k)$ such that $(w, \vec{0})$ is generated by G_M iff there are a vector $\vec{r} \in LOOP(M)$ and a loop-free derivation Δ of $(w, -\vec{r})$ in G with $M \subseteq N_\Delta$.

Finally, it is easy to construct a valence grammar G' with core rules of the forms $A \to BC$, $A \to a$, and $L(G') = \bigcup_{M \subseteq N} L(G_M)$. By Proposition 5.8, $L(G') = L(G)$. □

By Proposition 5.12, we may assume in the following that the core rules of a context-free valence grammar over $\mathbb{Z}^k$ are in Chomsky normal form.

5.5 Normalization of the valences

In order to prove the following proposition, we need another lemma:

Lemma 5.13 *Let $G = (N, T, P, S, \mathbb{Z}^k)$ be a valence grammar over $\mathbb{Z}^k$ so that the core rules of G are in Chomsky normal form. Then, for every length bound ℓ, there is an equivalent valence grammar $G'_\ell = (N', T, P', S', \mathbb{Z}^k)$ over $\mathbb{Z}^k$ such that any derivation $(S', \vec{0}) \overset{*}{\Rightarrow} (\xi, \vec{r})$ in G'_ℓ of some sentential form ξ with $|\xi| \leq \ell$ implies $\vec{r} = \vec{0}$. Moreover, the core rules of G'_ℓ are in Chomsky normal form, as well, and the new start symbol S' does not appear on the right-hand side of any rule in G'_ℓ.*

Proof. Let $N' = N \times E \cup \{S'\}$, where

$$E = \{\, \vec{r} \in \mathbb{Z}^k : \exists\xi, |\xi| \leq \ell, (S, \vec{0}) \overset{*}{\Rightarrow}_G (\xi, \vec{r}) \,\}.$$

Note that $\vec{0} \in E$ and $|E| < \infty$, since the underlying context-free grammar of G is assumed to be in Chomsky normal form. Consider, for each $\vec{r} \in E$, the function $f_{\vec{r}}$ which reads a sentential form ζ of G from left to right, outputs a when it reads a terminal symbol a, outputs $(A, \vec{r})$ the first time it reads some nonterminal symbol A of G and outputs $(A, \vec{0})$ when it encounters further nonterminals A. This means that

$$f_{\vec{r}}(\alpha_0 A_1 \ldots \alpha_{j-1} A_j \alpha_j) = \alpha_0 (A_1, \vec{r}) \alpha_1 (A_2, \vec{0}) \ldots \alpha_{j-1} (A_j, \vec{0}) \alpha_j$$

for terminal strings $\alpha_0, \ldots, \alpha_j$ and nonterminal symbols $A_1, \ldots, A_j$. Furthermore, let

$$L_\ell = \{\, (\xi, \vec{r}) \in (N \cup T)^* \times E : |\xi| \leq \ell, (S, \vec{0}) \overset{*}{\Rightarrow}_G (\xi, \vec{r}) \,\}.$$

It is quite easy to define a context-free (valence) grammar in Chomsky normal form (satisfying the requirements of the lemma) for the finite language $\{\, \alpha \in T^* : (\alpha, \vec{0}) \in L_\ell \,\}$. Let P'' be the rule set of this grammar. Now, define

$$\begin{array}{rcl} P' & = & \{\, (S' \to f_{\vec{r}}(\xi), \vec{0}) : (\xi, \vec{r}) \in L_\ell \wedge \xi \notin T^* \,\} \\ & \cup & \{\, ((A, \vec{r}) \to (B, \vec{0})(C, \vec{0}), \vec{r} + \vec{s}) : (A \to BC, \vec{s}) \in P \,\} \\ & \cup & \{\, ((A, \vec{r}) \to a, \vec{r} + \vec{s}) : (A \to a, \vec{s}) \in P \,\} \\ & \cup & P''. \end{array}$$

It is easy to verify that the grammar G'_ℓ constructed in this manner satisfies the claim. □

Proposition 5.14 *Let G be a valence grammar over $\mathbb{Z}^k$. Then, there is an equivalent valence grammar G' over $\mathbb{Z}^k$ with valence rules of the forms $(A \to BC, r \cdot \vec{e}_j), r \in \mathbb{Z}$, and $(A \to a, \vec{0})$.*

Proof. The case $k = 1$ is quite easy and is hence left to the reader. Therefore, let $k > 1$ in the following. Let $G = (N, T, P, S, \mathbb{Z}^k)$ be a valence grammar with core rules in Chomsky normal form according to Proposition 5.12. Moreover, let G satisfy the statement of the preceding lemma with $\ell = k+1$. Let r be large enough such that r upperbounds the maximum norm of any vector $\vec{r}$ which might be obtained as $(A,\vec{0}) \overset{*}{\Rightarrow}_G (\xi,\vec{r})$ for some sentential form ξ with $|\xi| \leq k+1$. Let $R \subset \mathbb{Z}^k$ be the finite set of all vectors whose maximum norm is at most r.

Consider $N \cup (N^{\leq k} \times R) \cup T'$ as new nonterminal alphabet, where T' consists of primed copies of the terminal alphabet T. Moreover, consider $'$ as a homomorphism which maps $a \in T$ into a' and $A \in N$ into A. Now add, for each sentential form $\xi \in (N \cup T)^{k+1}$ such that $(A,\vec{0}) \overset{*}{\Rightarrow}_G (\xi,\vec{r})$ with $\vec{r} = \sum_{j=1}^n r_j\vec{e}_j$ and $\xi = x_1 \dots x_{k+1}$, $x_i \in N \cup T$, the rules $(A \to x_1'(x_2 \dots x_{k+1}, \vec{r} - r_1\vec{e}_1), r_1\vec{e}_1)$ and $((x_j \dots x_{k+1}, \vec{r} - \sum_{i=1}^{j-1} r_i\vec{e}_i) \to x_j'(x_{j+1} \dots x_{k+1}, \vec{r} - \sum_{i=1}^{j} r_i\vec{e}_i), r_j\vec{e}_j)$ for $j = 2,\dots,k-1$ and $((x_k x_{k+1}, r_k\vec{e}_k) \to x_k' x_{k+1}', r_k\vec{e}_k)$.

Moreover, the rule set of G' contains the same start rules as G and termination rules $(a' \to a, \vec{0})$ for each $a \in T$.

The containment $L(G') \subseteq L(G)$ is obvious. Conversely, a word $w \in L(G)$ has a derivation of length $2|w|-1$ in G. If $|w| \leq k+1$, $w \in L(G')$ by a one-step derivation due to the construction of the preceding lemma. Otherwise, w can be derived in a leftmost derivation manner according to G as follows:

$$(S,\vec{0}) \overset{*}{\Rightarrow} (w_0A_0,\vec{0}) \overset{*}{\Rightarrow} (w_0w_1A_1,\vec{r_1}) \overset{*}{\Rightarrow} \dots (w = w_0w_1 \dots w_n, \vec{0} = \vec{r_n})$$

with $|w_0| \leq k$ and $|w_i| = k$ for $1 \leq i \leq n$. By construction, each of the involved subderivations

$$(A_i,\vec{0}) \overset{*}{\Rightarrow}_G (w_iA_{i+1}, \vec{r_i} - \vec{r_1})$$

for $0 \leq i < n$ can be simulated by a series of rule applications of G'. Moreover, the rule $(S \to w_0A_0, \vec{0})$ is in the rule set of G'. Therefore, we can conclude that $w \in L(G')$. □

Proposition 5.15 *Let G be a valence grammar over $\mathbb{Z}^k$. Then, there is an equivalent valence grammar G' over $\mathbb{Z}^k$ with valence rules of the forms $(A \to BC, r \cdot \vec{e}_j), r \in \{-1,0,1\}$, and $(A \to a, \vec{0})$.*

Proof. Let $G = (N,T,P,S,\mathbb{Z}^k)$ be of the form given in Proposition 5.14, and set $c = \max\{||\vec{r}||_1 : \exists(A \to \alpha, \vec{r}) \in P_2\}$. We construct $G' = (N',T,P',S')$ with

$$\begin{aligned}
N' &= N \times \{0,\dots,c-1\}^k, \\
S' &= (S,\vec{0}), \\
P' &= \{\, ((A,\vec{0}) \to a, \vec{0}) : (A \to a, \vec{0}) \in P \,\} \\
&\cup \{\, ((A,\vec{r}) \to (B,\vec{r_1})(C,\vec{r_2}), \vec{s}) : \\
&\qquad \vec{s} \in \{\vec{0}, \vec{e_j}, -\vec{e_j}\} \wedge \exists r_0(r_0 \in \{-c+1,\dots,c-1\} \wedge \\
&\qquad\quad (A \to BC, r_0\vec{e_j}) \in P \wedge c\vec{s} + \vec{r} = \vec{r_1} + \vec{r_2} + r_0\vec{e_j}) \,\}.
\end{aligned}$$

By bottom-up induction, it can be shown that $((A,\vec{r}),\vec{0}) \overset{*}{\Rightarrow}_{G'} (w,\vec{s})$ iff $\vec{r} \in \{-c+1,\dots,c-1\}^k$ and $(A,\vec{0}) \overset{*}{\Rightarrow}_G (w, c\vec{s}+\vec{r})$. In the induction step, one has to take care that *all* possible vectors $\vec{r} \in \{-c+1,\dots,c-1\}^k$ are covered. □

Proposition 5.16 *Let G be a valence grammar over $\mathbb{Z}^k$. Then, there is an equivalent valence grammar G' over $\mathbb{Z}^k$ with valence rules of the forms $(A \to BC, \vec{0})$, and $(A \to a, r \cdot \vec{e}_j), r \in \{-1,0,1\}$.*

Proof. Let $G = (N,T,P,S,\mathbb{Z}^k)$ have valence rules as in Proposition 5.15. Now, $G' = (N',T,P',S',\mathbb{Z}^k)$ with $N' = N \times E$, where $E = \{r \cdot \vec{e}_j : r \in \{-1,0,1\}, 1 \leq j \leq k\}$, $S' = (S,\vec{0})$, and

$$\begin{aligned} P' &= \{\, ((A,\vec{r}) \to (B,\vec{r})(C,\vec{s}),\vec{0}) : (A \to BC,\vec{s}) \in P, \vec{r} \in E \,\} \\ &\cup \{\, ((A,\vec{r}) \to a,\vec{r}) : (A \to a,\vec{0}) \in P, \vec{r} \in E \,\}. \end{aligned}$$

Again, it can easily be shown by bottom-up induction that $((A,\vec{r}),\vec{0}) \overset{*}{\Rightarrow}_{G'} (w,\vec{s})$ holds iff $(A,\vec{0}) \overset{*}{\Rightarrow}_G (w,\vec{s}+\vec{r})$ is true. □

5.6 Greibach normal form

Proposition 5.17 *Let $G = (N,T,P,S,\mathbb{Z}^k)$ be a valence grammar over $\mathbb{Z}^k$. There is an equivalent valence grammar $G' = (N',T,P',S',\mathbb{Z}^k)$ such that the rules of P' have the form $(A \to a\alpha, \vec{r})$, $A \in N', a \in T, \alpha \in N'^*, r \in \mathbb{Z}^k$, $||r||_1 \leq 1$.*

Proof. Let G be a valence grammar over $\mathbb{Z}^k$ with core rules in Chomsky normal form. Nearly literally the same construction as in the context-free case [11, Section 4.6] can be applied to obtain an equivalent valence grammar G' with core rules in Greibach normal form. The shape of the valences in G guarantees that during the construction, only valence rules of the forms $(A \to \alpha, \vec{0})$ or $(A \to a\alpha, \vec{r})$, $\|\vec{r}\|_1 \leq 1$, are produced. □

Remark 5.18 *With some additional effort, other normal forms, e.g., a quadratic double Greibach normal form, could be shown for $\mathcal{L}(\mathrm{Val},\mathrm{CF},\mathbb{Z}^k)$. The interested reader should study the corresponding construction in [1].*

6 An Iteration Lemma

We are going to prove iteration lemmas similar to the pumping lemmas for context-free and regular languages. We use the idea of minimal cycles, already present in Vicolov's proof of the strictness of the inclusions $\mathcal{L}(\mathrm{Val},\mathrm{CF},\mathbb{Z}^k) \subset$

$\mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbb{Z}^{k+1})$ [25]. We also need the normal form theorem from the previous section and the following elementary results:

On $\mathbb{N}^t$, let $\leq$ denote the natural partial order with $(a_1, \dots, a_t) \leq (b_1, \dots, b_t)$ iff $a_1 \leq b_1, \dots, a_t \leq b_t$. Instead of "$\vec{a} \leq \vec{b}$ and $\vec{a} \neq \vec{b}$", we simply write "$\vec{a} < \vec{b}$".

Lemma 6.1 (Dickson's Lemma) *Any infinite set $S \subseteq \mathbb{N}^t$, $t \geq 1$, contains two elements $\vec{a}, \vec{b}$ such that $\vec{a} < \vec{b}$.*

Lemma 6.2 *Let $A \in \mathbb{Z}^{k \times t}$ be a matrix with $t \geq k$. If the equation system $A \cdot \vec{x} = \vec{0}$ has a solution in $\mathbb{N}^t \setminus \{\vec{0}\}$, then it has a solution in $\mathbb{N}^t \setminus \{\vec{0}\}$ with at most $k+1$ positive components.*

Proof. Consider a solution $\vec{a} \in \mathbb{N}^t \setminus \{\vec{0}\}$ with more than $k+1$ positive components. Without loss of generality, let us assume that $\vec{a} = (a_1, \dots, a_t)$, with $a_1 > 0, \dots, a_s > 0$, $a_{s+1} = \dots = a_t = 0$, for some $s \geq k+2$. In $\mathbb{Z}^t \setminus \{\vec{0}\}$, the equation system has a solution $\vec{b} = (b_1, \dots, b_t)$, with $b_1 = b_{s+1} = \dots = b_t = 0$, and $b_r < 0$, for some $2 \leq r \leq s$. Let j be the index such that $b_j/a_j \leq b_i/a_i$, for all $1 \leq i \leq s$. This implies $b_j < 0$ and $a_j b_i - b_j a_i \geq 0$, for all $1 \leq i \leq s$. Then, $\vec{c} = (c_1, \dots, c_t) = -b_j \vec{a} + a_j \vec{b}$ is in $\mathbb{N}^t \setminus \{\vec{0}\}$ and has at most $s-1$ positive components, since $c_1 = -b_j a_1 > 0$, $c_i = -b_j a_i + a_j b_i \geq 0$ for $2 \leq i \leq s$, and $c_j = 0$, $c_i = 0$ for $s+1 \leq i \leq t$. By iteration, a solution with at most $k+1$ positive components is reached. $\square$

Given a valence grammar $G = (N, T, P, S, \mathbb{Z}^k)$ over $\mathbb{Z}^k$ (in normal form), a *cycle* is a derivation $(A, \vec{0}) \overset{*}{\Rightarrow} (vAw, \vec{r})$ with $A \in N, vw \in T^+$. A derivation is called *cycle-free* iff none of its subderivations is a cycle. A cycle is called *minimal* iff none of its proper subderivations is a cycle. For $M \subseteq N$, let $Z(M)$ be the set of all minimal cycles $\zeta = (A, \vec{0}) \overset{*}{\Rightarrow} (vAw, \vec{r})$ with $A \in M$.

Theorem 6.3 *For any infinite language $L \in \mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbb{Z}^k)$, $L \subseteq T^*$, there are a constant n and a finite set of iterative $(2k+2)$-tuples $I \subseteq (T^*)^{2k+2}$ such that:*

1. *$|\alpha_1 \dots \alpha_{2k+2}| > 0$, for all $(\alpha_1, \dots, \alpha_{2k+2}) \in I$, and*
2. *for all $w \in L$, $|w| \geq n$, there are a decomposition $w = z_1 z_2 \cdots z_{2k+2} z_{2k+3}$ and an iterative tuple $(\alpha_1, \dots, \alpha_{2k+2})$ such that*

$$z_1 \alpha_1^i z_2 \alpha_2^i \cdots z_{2k+2} \alpha_{2k+2}^i z_{2k+3} \in L \text{ for all } i \geq 0.$$

Proof. Let $G = (N, T, P, S, \mathbb{Z}^k)$ be a valence grammar over $\mathbb{Z}^k$ in Chomsky normal form generating L. The normal form guarantees that a derivation contains no (nontrivial) subderivations of the form $(A, \vec{0}) \overset{*}{\Rightarrow} (A, \vec{r})$. Given a derivation $\Delta : (S, \vec{0}) \overset{*}{\Rightarrow} (w, \vec{0})$, $w \in T^*$, we can successively erase minimal cycles to obtain a cycle-free derivation $\hat{\Delta} : (S,\vec{0}) \overset{*}{\Rightarrow} (\hat{w},\vec{r})$, $\hat{w} \in T^*$. Moreover, if $N_\Delta = M$ and $Z(M) = \{\zeta_1, \ldots, \zeta_t\}$, $\zeta_i : (A_i,\vec{0}) \overset{*}{\Rightarrow} (v_iA_iw_i,\vec{r_i})$, we can define a *deletion vector* $\vec{x}(\Delta) \in \mathbb{N}^t$ whose ith component is the number of times that the minimal cycle ζ_i is deleted in the above process.

Now let N_{inf} be the set of all $M \subseteq N$ with $N_\Delta = M$, for infinitely many derivations of the form $\Delta : (S,\vec{0}) \overset{*}{\Rightarrow} (w,\vec{0})$, $w \in T^*$, and consider some $M \in N_{inf}$.

Since the number of cycle-free derivations is finite, there are infinitely many derivations Δ_i, $i = 1, 2, \ldots$, such that $\hat{\Delta}_i = \hat{\Delta}_1$, for all $i \geq 1$ and $N_\Delta = M$. Moreover, the set $\{\vec{x}_i : \vec{x}_i = \vec{x}(\Delta_i), i \geq 1\}$, is infinite. By Lemma 6.1, there are two indices j, k such that $\vec{x}_j < \vec{x}_k$. Let $\mathfrak{Z} \in \mathbb{Z}^{k\times t}$ be the matrix whose columns are the valences of the derivations $\zeta_1, \ldots, \zeta_t$ (in this sequence). Clearly, all $\vec{x}_i$, $i \geq 1$, are solutions of $\mathfrak{Z} \cdot \vec{x} = -\mathrm{val}(\hat{\Delta}_1)$. Consequently, $\vec{y} = \vec{x}_k - \vec{x}_j$ is a solution of $\mathfrak{Z} \cdot \vec{x} = \vec{0}$ with $\vec{y} \in \mathbb{N}^t \setminus \{\vec{0}\}$. By Lemma 6.2, there is vector $\vec{a} = (a_1, \ldots, a_t) \in \mathbb{N}^t \setminus \{\vec{0}\}$ with at most $k+1$ positive components and $\mathfrak{Z} \cdot \vec{a} = \vec{0}$.

Let us assume that $a_{k+2} = a_{k+3} = \ldots = a_t = 0$ (otherwise, rearrange the minimal cycles in $Z(M)$). A derivation $\Delta : (S,\vec{0}) \overset{*}{\Rightarrow} (z,\vec{0})$, $z \in T^*$ can now be extended to derivations $(S,\vec{0}) \overset{*}{\Rightarrow} (z_i,\vec{0})$, $z_i \in T^*$, $i \in \mathbb{N}$, by inserting the cycles $\zeta_1^{a_1\cdot i}, \ldots, \zeta_{k+1}^{a_{k+1}\cdot i}$ at appropriate places. (For a cycle $\zeta : (A,\vec{0}) \overset{*}{\Rightarrow} (vAw,\vec{r})$, ζ^m is defined as the cycle $(A,\vec{0}) \overset{*}{\Rightarrow} (v^mAw^m, m\cdot\vec{r})$.)

With respect to M, we obtain the set of iterative $(2k+2)$-tuples $I(M)$ as the set of permutations of $\{v_1^{a_1}, w_1^{a_1}, \ldots, v_{k+1}^{a_{k+1}}, w_{k+1}^{a_{k+1}}\}$. (In fact, only certain permutations are really possible.) The set of iterative tuples I mentioned in the theorem is found as $I = \bigcup_{M\in N_{inf}} I(M)$, and the constant n from the theorem is

$$n = \max\{|w| : w \in T^* \wedge \exists\Delta(\Delta : (S,\vec{0}) \overset{*}{\Rightarrow} (w,\vec{0}) \wedge N_\Delta \notin N_{inf})\}.$$

$\square$

In the case of regular valence languages, we can analogously show (now using minimal cycles of the form $(A,\vec{0}) \overset{*}{\Rightarrow} (wA,\vec{r})$):

Theorem 6.4 *For any infinite language $L \in \mathcal{L}(\mathrm{Val}, \mathrm{REG}, \mathbb{Z}^k)$, $L \subseteq T^*$, there are a constant n and a finite set of iterative $(k+1)$-tuples $I \subseteq (T^*)^{k+1}$ such that:*

1. *$|\alpha_1 \ldots \alpha_{k+1}| > 0$, for all $(\alpha_1, \ldots, \alpha_{k+1}) \in I$, and*

2. *for all $w \in L$, $|w| \geq n$, there are a decomposition $w = z_1z_2\cdots z_{k+1}z_{k+2}$ and an iterative tuple $(\alpha_1, \ldots, \alpha_{k+1})$ such that*

$$z_1\alpha_1^i z_2\alpha_2^i \cdots z_{k+1}\alpha_{k+1}^i z_{k+2} \in L \text{ for all } i \geq 0. \quad \square$$

Finally, we give a refinement of the last two theorems, stating that particular cycles appear in some iteration. We apply this result in [6] to show that a certain type of parallel valence systems (ET0L systems with table valences) cannot produce languages with arbitrarily fast growing length sets.

For a derivation Δ and a cycle ζ, let $n_\zeta(\Delta)$ be the number of appearances of the subderivation ζ in Δ.

Corollary 6.5 *Let $G = (N, T, P, S, \mathbb{Z}^k)$ be a context-free valence grammar producing an infinite language. Let $\zeta = (A, \vec{0}) \overset{*}{\Rightarrow} (vAw, \vec{r})$ be a minimal cycle such that*

$$\{n_\zeta(\Delta) : \Delta : (S, \vec{0}) \overset{*}{\Rightarrow} (u, \vec{0}), u \in T^*\}$$

is unbounded. Then, there is a word $z \in L(G)$ such that some iterative tuple applicable on z has the form

$$(\alpha_1, \ldots, \alpha_{2k+2}), \text{ with } \alpha_i = v^m, \alpha_j = w^m \text{ for some } 1 \le i < j \le 2k+2, m \ge 0.$$

Proof. As $L(G)$ is infinite and N is finite, there has to be a subset $M \subseteq N$ such that

$$\{n_\zeta(\Delta) : \Delta : (S, \vec{0}) \overset{*}{\Rightarrow} (u, \vec{0}), u \in T^*, N_\Delta = M\}$$

is unbounded. Inspecting the proof of Theorem 6.3, we can order $Z(M)$ such that $\zeta_1 = \zeta$, and find an infinite sequence of deletion vectors $\vec{x_1}, \vec{x_2}, \ldots$ such that the first component is strictly growing. By Lemma 6.1, there are $i < j$ such that $\vec{x_i} < \vec{x_j}$. The vector $\vec{x} = \vec{x_j} - \vec{x_i}$ is in $\mathbb{N}^t \setminus \{\vec{0}\}$, has a positive first component, and satisfies $\mathfrak{Z}\vec{x} = \vec{0}$. When constructing the solution with at most $k+1$ positive components as indicated in the proof of Lemma 6.2, the first component is not changed. □

Completely analogously, we can derive:

Corollary 6.6 *Let $G = (N, T, P, S, \mathbb{Z}^k)$ be a regular valence grammar producing an infinite language. Let $\zeta = (A, \vec{0}) \overset{*}{\Rightarrow} (vA, \vec{r})$ be a minimal cycle such that*

$$\{n_\zeta(\Delta) : \Delta : (S, \vec{0}) \overset{*}{\Rightarrow} (u, \vec{0}), u \in T^*\}$$

is unbounded. Then, there is a word $z \in L(G)$ such that some iterative tuple applicable on z has the form

$$(\alpha_1, \ldots, \alpha_{k+1}), \text{ with } \alpha_i = v^m \text{ for some } 1 \le i \le k+1, m \ge 0. \quad \square$$

7 Conclusions and perspectives

We have given an overview of the potentials of (sequential) valence grammars. Many problems remain open. It would, for instance, be very interesting to investigate valence grammars over other specific monoids than discussed here in order to describe different language classes. Another interesting issue could be the investigation of deterministic valence automata and their characterizations in terms of valence grammars. Of course, one has to be aware of the close equivalences to k-*BLIND* and reversal-bounded multi-counter machines and their deterministic variants, see [9, 10]. The latter question could be very interesting keeping in mind the correspondence of Chomsky normal form grammars and machine models, since we can hope for efficient LR-style parsing algorithms for valence grammars. Valences can be used in this way as attributes which can be easily handled.

As pointed out in Section 4.3, it is of interest to discuss valence grammars accepting with a finite set of monoid elements. With this modified definition, the equivalence of valence grammars over finite monoids and matrix grammars can be established. We will discuss this issue in a forthcoming paper.

An interesting variant of valence grammars are grammars with valuations, where values are assigned to terminals instead of rules. In the case of commutative monoids, these languages are homomorphic pre-images of valence languages.

Finally, we would like to thank our colleagues, especially H. Bordihn, M. Holzer, K. Reinhardt, L. Staiger and D. Thérien for stimulating discussions.

References

[1] J.-M. Autebert, J. Berstel and L. Boasson. Context-free languages and pushdown automata. In: G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, Vol. 1*, pages 111–174. Berlin: Springer, 1997.

[2] J. Dassow and Gh. Păun. *Regulated Rewriting in Formal Language Theory.* Berlin: Springer, 1989.

[3] J. Dassow, Gh. Păun and A. Salomaa. Grammars with controlled derivations. In: G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, Vol. 2*, pages 101–154. Berlin: Springer, 1997.

[4] H. Fernau, R. Freund and M. Holzer. Hybrid modes in cooperating distributed grammar systems: internal versus external hybridization. *Theoretical Computer Science*, to appear 2001.

[5] H. Fernau and R. Stiebe. Regulation by valences. In: I. Prívara and P. Ružička, editors, *Mathematical Foundations of Computer Science 1997 (LNCS 1295)*, pages 239–248, 1997. Berlin: Springer, 1997. An extension appeared as Technical Report TR-36 of Martin-Luther-Universität Halle-Wittenberg, Fachbereich Mathematik und Informatik, 1998.

[6] H. Fernau and R. Stiebe. Valences in Parallel Systems. Accepted for publication in *Fundamenta Informaticae*, 2000. Also see: Technical Report TR-18 of Martin-Luther-Universität Halle-Wittenberg, Fachbereich Mathematik und Informatik, 2000.

[7] H. Fernau and R. Stiebe. Remarks on Regulated Valence Grammars. Unpublished Manuscript, 2000.

[8] M. Gheorge. Linear valence grammars. In: *International Meeting of Young Computer Scientists*, pages 281–285, 1986.

[9] S. Greibach. Remarks on blind and partially blind one-way multicounter machines. *Theoretical Computer Science*, 7:311–324, 1978.

[10] E. M. Gurari and O. H. Ibarra. The complexity of decision problems for finite-turn multicounter machines. *Journal of Computer and System Sciences*, 22:220–229, 1981.

[11] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Reading (MA): Addison-Wesley, 1979.

[12] O. H. Ibarra, S. K. Sahni, and C. E. Kim. Finite automata with multiplication. *Theoretical Computer Science*, 2:271–296, 1976.

[13] M. Marcus and Gh. Păun. Valence gsm-mappings. *Bulletin Mathématique de la Societé de Science Math. de Roumanie*, 31(3):219–229, 1987.

[14] V. Mitrana. Valence grammars on a free generated group. *EATCS Bulletin*, 47:174–179, 1992.

[15] V. Mitrana and R. Stiebe. The accepting power of finite automata over groups. In Gh. Păun and A. Salomaa, editors, *New Trends in Formal Languages (LNCS 1218)*, pages 39–48. Berlin: Springer, 1997.

[16] Gh. Păun. A new generative device: valence grammars. *Revue Roumaine de Mathématique Pures et Appliquées*, XXV(6):911–924, 1980.

[17] Gh. Păun. On a class of valence grammars. *Stud. cerc. mat.*, 42:255–268, 1990.

[18] Gh. Păun. Valences: increasing the power of grammars, transducers, grammar systems. *EATCS Bulletin*, 48:143–156, 1992.

[19] V. Red'ko and L. Lisovik. Regular events in semigroup. *Problems of Cybernetics*, 37:155–184, 1980. In Russian.

[20] J. J. Rotman. *An Introduction to the Theory of Groups.* New York: Springer, 5th edition, 1995.

[21] G. Rozenberg and A. K. Salomaa. *The Mathematical Theory of L Systems.* Academic Press, 1980.

[22] A. K. Salomaa. *Formal Languages.* Academic Press, 1973.

[23] G. Satta. The membership problem for unordered vector grammars. In J. Dassow, G. Rozenberg, and A. Salomaa, editors, *Developments in Language Theory II*, pages 267–275. Singapore: World Scientific, 1996.

[24] I. H. Sudborough. The complexity of the membership problem for some extensions of context-free languages. *International Journal of Computer Mathematics*, 6:191–215, 1977.

[25] S. Vicolov. Hierarchies of valence languages. In J. Dassow and A. Kelemenova, editors, *Developments in Theoretical Computer Science*, pages 191–196. Basel: Gordon and Breach, 1994.

[26] S. Vicolov-Dumitrescu. Grammars, grammar systems, and gsm mappings with valences. In Gh. Păun, editor, *Mathematical Aspects of Natural and Formal Languages*, pages 473–491. Singapore: World Scientific, 1994.