Valences in Parallel Systems

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`

Abstract

We investigate four variants of valence regulations incorporated in (limited) Lindenmayer systems, focussing on hierarchical relations and closure properties. Strong connections to well-known families of languages obtained by regulated rewriting are established.

We prove several new results on valence transducers and valence generalized sequential machines (gsm). In this way, we can show new properties of ET0L languages, namely the non-closure under permutation, quasiintersection, valence gsm mappings and intersection with regular valence languages. Moreover, we solve a question marked as open in [32] by proving that the uniformly k-limited ET0L languages form a full semi-AFL for all $k \geq 1$.

Contents

1 Introduction

Valence grammars, introduced by Gh. Păun in [22], are simple generalizations of Chomsky grammars. To any rule, a so-called valence is added, which is a positive rational number. Valences are extended to derivations by multiplying the valences of the applied rules, and only terminal derivations with a valence of 1 are allowed. Up to now, the idea of valences has been studied only in relation with sequential rewriting. There are several natural extensions of valences to parallel systems. We shall consider the following variants of valence Lindenmayer systems:[1]

- In valence ET0L systems, a derivation of a string is admissible if the derivation's valence product evaluates to 1.
- In valence test ET0L systems, a derivation of a string is admissible if the valence product of each table application during the derivation evaluates to 1.
- In table valence ET0L system, values are given to tables instead of symbols as in the first two cases. A derivation of a string is admissible if the valence product of all tables which are applied during the derivation evaluates to 1.

Moreover, we consider valuated ET0L systems. Here, valences are assigned to symbols and, as morphic extension, to strings, and a derivation of a string is admissible if the string's valence product evaluates to 1.

We advice the reader not to confuse our concepts either with the concept of *grammars with valuations* as introduced in [2] or with that of *evaluated grammars* presented in [21].

Besides being an interesting and motivated topic in its own right, the study of variants of ET0L systems with valences furthers the investigations on regulated parallel systems, cf. [4, Chapter 8] for regulated (E)T0L systems and [14, 27, 28, 29, 30] for regulated limited (E)T0L systems. More precisely, the way regulation mechanisms are considered in parallel systems corresponds to our approach of ET0L systems with table valences. Two other variants give some internal control on the application of single rules. Observe that most other internal control mechanisms (borrowed from the area of sequential regulated rewriting, see [4]) do not enhance the power of ET0L systems (with random context ET0L systems as notable exception), in contrast to the valence (test) systems considered here. So, this paper might also revive interest in regulated parallel systems in general.

[1]In this introductory discussion, we assume some knowledge on Lindenmayer systems on the side of the reader. Definitions will be given in later sections. ET0L systems are parallel rewriting systems with **E**xtensions, i.e., a designated terminal alphabet, and **T**ables, cf. [24].

The concept of valuated systems is somehow independent of the underlying derivation mechanism and could be added to any language describing device, which opens a new area of formal language studies.

We prove several new results on valence transducers and valence general sequential machines (gsm). In this way, we can show new properties of ET0L languages, namely the non-closure under permutation, quasiintersection, valence gsm mappings and intersection with regular valence languages. Moreover, we solve a question marked as open in [32] by proving that the uniformly k-limited ET0L languages form a full semi-AFL for all $k \geq 1$.

The paper is organized as follows: First, we provide the necessary definitions for (limited) ET0L systems regulated by valences. In Section 3, we exhibit close relationships between valence test systems and programmed grammars. Section 4 shows that there are also connections between programmed grammars and valence or valuated (limited) ET0L systems, respectively. In Section 5, we discuss the generative power and complexity of ET0L systems where the valences are attached to tables. In Section 6, we are interested in normal form representations of ET0L systems with valences in two senses: first, we show that T0L systems are nearly as powerful as ET0L systems, and then we show that the hierarchy defined via the degree of synchronization (i.e., the number of tables) of such systems collapses to the second (and sometimes even to the first) level. In Section 7, closure properties of the corresponding language families are investigated, implying also some results concerning the generative power. We conclude the paper with a list of open questions and possible future research directions. Many (other) open problems are indicated where appropriate, throughout the paper.

Parts of this paper have been presented at MFCS'97 conference, see [12].

2 Definitions

Let $\mathbb{Q}_+$ denote the set of positive rational numbers. We mostly consider the multiplicative monoid $(\mathbb{Q}_+, \cdot, 1)$.

A *valence ET0L system* (over $(\mathbb{Q}_+, \cdot, 1)$) is a quadruple $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$, where Δ is a nonempty subset of the alphabet Σ, $\omega \in \Sigma^+$, and $\mathfrak{P}$ is a finite set of *tables* $\mathfrak{P} = \{P_1, \ldots, P_k\}$ where each P_i, $1 \leq i \leq k$ is a finite subset of $\Sigma \times \Sigma^* \times \mathbb{Q}_+$ satisfying the condition that, for each $a \in \Sigma$, there are $w \in \Sigma^*, r \in \mathbb{Q}_+$ such that $(a, w, r) \in P_i$. As usual, we write $(a \to w, r) \in P_i$ instead of $(a, w, r) \in P_i$.

G is called a *valence T0L system* if $\Sigma = \Delta$; if the number of tables is one, then G is a *valence E0L system*; if G is both a valence T0L and a valence E0L system it is called a *valence 0L system*. Furthermore, we inherit the notion of propagating and of deterministic valence ET0L system from the theory of ET0L systems, abbreviated as EPT0L and EDT0L system, respectively:

- $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$ is called *propagating* if each table $P \in \mathfrak{P}$ satisfies $P \subseteq \Sigma \times \Sigma^+ \times \mathbb{Q}_+$.
- $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$ is called *deterministic* if

$$\forall P \in \mathfrak{P}\ \forall a \in \Sigma((a, w_1, q_1), (a, w_2, q_2) \in P \Rightarrow w_1 = w_2 \wedge q_1 = q_2)\,.$$

The yield relation $\Rightarrow$ is defined in the following way: $(w, r) \Rightarrow (w', r')$, with $w, w' \in \Sigma^*$, $r, r' \in \mathbb{Q}_+$ iff

1. $w = a_1 a_2 \ldots a_m, a_i \in \Sigma, 1 \leq i \leq m$,
2. $w' = w_1 w_2 \ldots w_m, w_i \in \Sigma^*, 1 \leq i \leq m$,
3. $r' = r_1 \cdot r_2 \cdot \ldots \cdot r_m \cdot r, 1 \leq i \leq m$, and
4. there is a table $P \in \mathfrak{P}$ such that $(a_i \to w_i, r_i) \in P$, for each $1 \leq i \leq m$.

The language generated by G is given by $L(G) = \{w \in \Delta^* \mid (\omega, 1) \stackrel{*}{\Rightarrow} (w, 1)\}$.

A *valence test ET0L system* (over $(\mathbb{Q}_+, \cdot, 1)$) looks formally like a valence ET0L system, i.e., it is a quadruple $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$. The yield relation $\Rightarrow$ is defined in the following different way: $w \Rightarrow w'$, where $w, w' \in \Sigma^*, r, r' \in \mathbb{Q}_+$ iff

1. $w = a_1 a_2 \ldots a_m, a_i \in \Sigma, 1 \leq i \leq m$,
2. $w' = w_1 w_2 \ldots w_m, w_i \in \Sigma^*, 1 \leq i \leq m$,
3. $1 = r_1 \cdot r_2 \cdot \ldots \cdot r_m, 1 \leq i \leq m$, and
4. there is a table $P \in \mathfrak{P}$ such that $(a_i \to w_i, r_i) \in P$, for each $1 \leq i \leq m$.

The language generated by G is given by $L(G) = \{w \in \Delta^* \mid \omega \stackrel{*}{\Rightarrow} w\}$. The several variants of valence test ET0L systems are defined analogously.

A *table valence ET0L system* (over $(\mathbb{Q}_+, \cdot, 1)$) is a quintuple $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$, where $G' = (\Sigma, \Delta, \mathfrak{P}, \omega)$ is an ET0L system, and $\varphi : \mathfrak{P} \to \mathbb{Q}_+$ is a mapping. The yield relation $\Rightarrow$ is defined in the following way: $(w, r) \Rightarrow (w', r')$, where $w, w' \in \Sigma^*$, $r, r' \in \mathbb{Q}_+$ iff $w \Rightarrow_{G'} w'$ using the table $P \in \mathfrak{P}$ and $r' = r \cdot \varphi(P)$. The language generated by G is given by $L(G) = \{w \in \Delta^* \mid (\omega, 1) \stackrel{*}{\Rightarrow} (w, 1)\}$. The several variants of table valence ET0L systems are defined analogously.

A *valuated ET0L system* (over $(\mathbb{Q}_+, \cdot, 1)$) is a quintuple $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ where $G' = (\Sigma, \Delta, \mathfrak{P}, \omega)$ is an ET0L system, and $\varphi : \Delta \to \mathbb{Q}_+$ is the valuation mapping. φ is extended as monoid morphism by $\varphi(\lambda) = 1, \varphi(aw) = \varphi(a) \cdot \varphi(w)$, for $a \in \Delta, w \in \Delta^*$. The language generated by G is given by $L(G) = \{w \in \Delta^* \mid \omega \stackrel{*}{\Rightarrow} w \wedge \varphi(w) = 1\}$. The several variants of valuated ET0L systems are defined analogously.

Let the language families generated by valence, valence test, table valence, and valuated, respectively, [E][P][D][T]0L systems be denoted by $\mathcal{L}(\mathrm{Val}, [\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L})$, $\mathcal{L}(\mathrm{Val}_t, [\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L})$, $\mathcal{L}([\mathrm{E}][\mathrm{P}][\mathrm{D}](\mathrm{Val})[\mathrm{T}]0\mathrm{L})$, and $\mathcal{L}(\mathrm{Val}', [\mathrm{E}][\mathrm{P}][\mathrm{D}]\mathrm{T}0\mathrm{L})$, respectively.

We also briefly consider valence, valence test and valuated, respectively, (uniformly) k-limited [E][P][D][T]0L systems and denote the corresponding language classes by $\mathcal{L}(\mathrm{Val}, (u)k\ell[\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L})$, $\mathcal{L}(\mathrm{Val}_t, (u)k\ell[\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L})$, and $\mathcal{L}(\mathrm{Val}', (u)k\ell[\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L})$, respectively. In contrast to the totally parallel derivation encountered in ET0L systems, in k-limited systems, exactly k occurrences of each symbol are replaced (if less than k occurrences are contained in the present string, all of them are replaced) in each derivation step. In uniformly k-limited systems (this gives us the additional letter u in our language class denotations), exactly k occurrences of symbols are replaced (if the present string is shorter than k, all symbols are replaced) in each derivation step.

In all variants of valence and valuated L systems, the set of prime numbers appearing in the factorizations of the valences or valuations is of importance. It will be denoted by $\mathrm{Primes}(G)$, for any such system G. In the literature on valence grammars, also additive integer vector valences are considered. Note that a (multiplicative) valence system G over $\mathbb{Q}_+$ can be seen as an additive valence system over $\mathbb{Z}^k$ where $k = |\mathrm{Primes}(G)|$. In what follows we shall always choose the more convenient representation. Results on additive valence systems over $\mathbb{Z}$ (i.e., multiplicative system with only one prime number in the factorizations) will be mentioned only as an aside, denoting the language classes by, e.g., $\mathcal{L}(\mathrm{Val}_t, [\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L}, \mathbb{Z})$.

From the theory of sequential regulated rewriting, we need the notion of a *programmed grammar*. The rules of such grammars are triples $(r : \alpha \to \beta, \sigma(r), \varphi(r))$ where $r : \alpha \to \beta$ is a rewriting rule labelled by r, the success field $\sigma(r)$ and the failure field $\varphi(r)$ are sets of labels. If, during a derivation, the rule labelled r is chosen and applicable, it is applied, and the next rule is selected from $\sigma(r)$. If the chosen rule is not applicable, the sentential form is unchanged, and a rule from $\varphi(r)$ is selected. If $\varphi(r) = \emptyset$ for all r, the programmed grammar is called *programmed grammar without appearance checking.* For the complete definitions, see [4]. The language families generated by programmed grammars and by programmed grammars without appearance checking with context-free [non-erasing] rules are denoted by $\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda], ac)$ and by $\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda])$. In one place, we also mention programmed languages with valence rules, see [11].

The set of symbols actually appearing in a word w is denoted by $\mathrm{Alph}(w)$. $\mathrm{Perm}(w)$ ($\mathrm{Perm}(L)$) denotes the set of permutations of word w (of language L, respectively), see [4].

3 The power of valence tests

3.1 Lindenmayer systems

It is well-known from the theory of Lindenmayer systems that propagating and non-propagating extended systems basically lead to the same language class. This is not true anymore in the case of valence test ET0L systems, as the following theorem shows (using well-known facts on regulated rewriting, see [4]). In order to establish our result, we need the following lemma.

Lemma 3.1 Let $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{Z}^n$ be given. Then there exists an effectively constructible bound $c = c(\vec{v}_1, \ldots, \vec{v}_k)$ such that, for every k-tuple of natural numbers $(a_1, \ldots, a_k)$ with

$$a_1 \cdot \vec{v}_1 + \ldots + a_k \cdot \vec{v}_k = \vec{0}\,, \tag{1}$$

there exists a sequence of indices $j_1, \ldots, j_m$ with $m = a_1 + \cdots + a_k$ such that $|\{i \mid j_i = j\}| = a_j$, for $1 \leq j \leq n$, and

$$\|\sum_{i=1}^{\ell} \vec{v}_{j_i}\| \leq c\,, \text{ for all } 1 \leq \ell \leq m\,. \tag{2}$$

(By $\|\vec{v}\|$ we denote the maximum norm of $\vec{v} \in \mathbb{Z}^n$: $\|\vec{v}\| = \max\{|v_i| : 1 \leq i \leq n\}$, for $\vec{v} = (v_1, \ldots, v_n)$.)

Proof. Given $\vec{v}_1, \ldots, \vec{v}_k$, we define the alphabet $\Sigma = \{c_1, \ldots c_k\}$ with valuation $\varphi(c_i) = \vec{v}_i$, $1 \leq i \leq k$, the Parikh mapping $\Psi : \Sigma^* \to \mathbb{N}^k$ with $\Psi(c_i) = \vec{e}_i$, $1 \leq i \leq k$, and the language $L = \{w \in \Sigma^* \mid \varphi(w) = \vec{0}\} \cap c_1^* \cdots c_k^*$. For any $w \in c_1^* \cdots c_k^*$ with $\Psi(w) = (a_1, \ldots, a_k)$, $a_1 \cdot \vec{v}_1 + \ldots + a_k \cdot \vec{v}_k = \varphi(w)$ holds. Hence, a word $w \in c_1^* \cdots c_k^*$ is in L iff its Parikh vector is a solution of equation (1).

Since L is a bounded regular valence language, it has a semilinear Parikh set which is effectively constructible, see [18, Theorem 3.4]. Hence, we can construct a regular language R which is Parikh equivalent to L. Now a k-tuple $(a_1, \ldots, a_k)$ is a solution of (1) iff R contains a word with Parikh vector $(a_1, \ldots, a_k)$. Let $\mathcal{A}$ be a deterministic finite automaton accepting R and containing only states that can be transferred to accepting states. If two words are sent to the same state by $\mathcal{A}$ then their valuations must be equal. Thus, any state of $\mathcal{A}$ is associated to a valuation. Let c be the largest maximum norm of these valuations. For any word $w \in R$ and for all prefixes v of w, we have $\|\varphi(v)\| \leq c$. □

Theorem 3.2 1. $\mathcal{L}(\mathrm{RE}) = \mathcal{L}(\mathrm{Val}_t, \mathrm{ET0L}) = \mathcal{L}(\mathrm{Val}_t, \mathrm{ET0L}, \mathbb{Z})$;

2. $\mathcal{L}(\mathrm{P}, \mathrm{CF}-\lambda, \mathrm{ac}) = \mathcal{L}(\mathrm{Val}_t, \mathrm{EPT0L}) = \mathcal{L}(\mathrm{Val}_t, \mathrm{EPT0L}, \mathbb{Z})$.

Proof. Since $\mathcal{L}(\mathrm{RE}) = \mathcal{L}(\mathrm{P}, \mathrm{CF}, \mathrm{ac})$, we have only to show how to simulate programmed grammars by valence test ET0L systems and vice versa.

All the inclusions from right to left are quite obvious but one. Therefore, we only sketch the idea how to simulate a valence test ET0L system by a programmed grammar with appearance checking. Firstly, consider the simulation of ET0L systems without valences by programmed grammars. The programmed grammar applies the rules of one parallel step in an arbitrary sequence. It uses disjoint alphabets for left-hand and right-hand sides and appearance checking to guarantee that each symbol is replaced. In the case of valence tests, one has to keep track of the valences. By the lemma above, one can restrict oneself to those sequential derivations whose valence norms never exceed the threshold c constructed in the proof of the preceding lemma.

On the other hand, let $G = (N, T, P, S)$ be a programmed grammar with appearance checking. Let $\mathrm{Lab}(P) = \{p_1, \ldots, p_r\}$ with $(p_i : A_i \to w_i, \sigma_i, \varphi_i)$. Let p be a prime number. We give a simulating valence test ET0L system $G' = (N \cup T \cup \{X, S'\} \cup \mathrm{Lab}(P), T, \mathfrak{P}, S')$. X is a special failure symbol. For each p_i, we have two simulating tables:

1. one is responsible for the appearance checking case, i.e.,

$$\begin{aligned} P_{i,ac} &= \{(p_i \to q, 1) \mid q \in \varphi_i\} \cup \\ &\cup \{(B \to B, 1) \mid B \in (N \setminus \{A_i\}) \cup T \cup \{X\}\} \cup \\ &\cup \{(B \to X, 1) \mid B \in \{A_i, S'\} \cup (\mathrm{Lab}(P) \setminus \{p_i\})\}, \end{aligned}$$

2. and one is responsible for the non-appearance checking case, i.e.,

$$\begin{aligned} P_i &= \{(A_i \to w_i, p)\} \cup \{(p_i \to q, p^{-1}) \mid q \in \sigma_i\} \cup \\ &\cup \{(B \to B, 1) \mid B \in N \cup T \cup \{X\}\} \cup \\ &\cup \{(B \to X, 1) \mid B \in \{S'\} \cup (\mathrm{Lab}(P) \setminus \{p_i\})\}. \end{aligned}$$

In addition, we have one initialization and one termination table:

$$\begin{aligned} \mathrm{init} &= \{(S' \to Sq, 1) \mid q \in \mathrm{Lab}(P)\} \cup \{(B \to X, 1) \mid B \neq S'\}, \\ \mathrm{exit} &= \{(a \to a, 1) \mid a \in T\} \cup \{(q \to \lambda, 1) \mid q \in \mathrm{Lab}(P)\} \cup \\ &\cup \{(B \to X, 1) \mid B \in N \cup \{X, S'\}\}. \end{aligned}$$

Define $\mathfrak{P} = \{P_{i,ac}, P_i \mid 1 \le i \le r\} \cup \{\text{init}, \text{exit}\}$.

In the case of propagating systems, we can exploit the closure of programmed languages under derivatives and the easily shown closure of valence test EPT0L languages under union in order to use nearly the same construction to prove the claim. □

Open Problem 1. We conjecture that deterministic systems are weaker than nondeterministic ones in the case of valence test E[P]T0L systems, but we could not prove this. □

Remark 3.3 By the trivial inclusion of so-called multiplicative valence level grammars within valence test E[P]0L languages, the inclusion shown in [23, Theorem 4] is a simple consequence. Moreover, if the inclusion $\mathcal{L}(\mathrm{Val}_t, \mathrm{EP0L}) \subseteq \mathcal{L}(\mathrm{Val}_t, \mathrm{EPT0L})$ can be shown to be strict, an open problem posed on [23, page 265] would be solved.

Remark 3.4 By an easy modification of the proof of the preceding theorem, the inclusion

$$\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda]) \subset \mathcal{L}(\mathrm{Val}_t, \mathrm{E[P]0L})$$

can be shown. Namely, we use a special prime number to control which productions have to be selected in one table application; unfortunately, the appearance checking feature is lost, which is usually trivial when dealing with Lindenmayer systems. This inclusion is strict, since the 0L language

$$\{a^{2^n} \mid n \ge 0\}$$

is not in $\mathcal{L}(\mathrm{P}, \mathrm{CF})$ by [17]. The classes $\mathcal{L}(\mathrm{Val}_t, \mathrm{E[P]0L}) = \mathcal{L}(\mathrm{Val}, \mathrm{E[P]0L})$ deserve further studies.

3.2 Limited systems

Literally the same proof as given in the preceding theorem works in the case of simulating programmed grammars via valence test k-limited ET0L systems. The adaptation of the simulation of k-limited ET0L systems (without valences) by programmed grammars is easier because only a bounded number of sequential derivation steps is needed for one parallel step. We leave the details to the reader.

Theorem 3.5 Let $k \in \mathbb{N}$. Then, we can show

1. $\mathcal{L}(\mathrm{RE}) = \mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{ET0L}) = \mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{ET0L}, \mathbb{Z})$, and

2. $\mathcal{L}(\mathrm{P}, \mathrm{CF}-\lambda, \mathrm{ac}) = \mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{EPT0L}) = \mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{EPT0L}, \mathbb{Z})$. □

Observe that it is not known whether k-limited ET0L systems are as powerful as valence test k-limited ET0L systems, cf. [6, 7, 14, 26]. The comments following Theorem 3.2 concerning determinism and one-table-restriction apply here, too; as we will later see, k-limited valence test ET0L systems with one table even characterize $\mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{E[P]T0L})$, cf. Theorem 6.3.

Finally, we consider uniformly k-limited ET0L systems. Here, we have to distinguish two cases.

Theorem 3.6 Let $k \in \mathbb{N}$.

- If $k = 1$, we have
  $\mathcal{L}(\mathrm{CF}) = \mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{E[P]T0L}) = \mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{E[P]T0L}, \mathbb{Z})$.

- Let $k \geq 2$. Then, we find:

  1. $\mathcal{L}(\mathrm{P}, \mathrm{CF}) = \mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{ET0L}) = \mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{ET0L}, \mathbb{Z})$, and
  2. $\mathcal{L}(\mathrm{P}, \mathrm{CF}-\lambda) = \mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{EPT0L}) = \mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{EPT0L}, \mathbb{Z})$.

Proof. Let us briefly discuss the case where $k = 1$. Since (at most) one production is applied when applying a table, this rule must carry the valence 1. Therefore, we may assume without loss of generality that all valences equal 1. In other words, we make no use of the valences at all. The claim follows with [32, Theorem 2.2].

When simulating a given valence test uniformly k-limited ET0L system by a programmed grammar, we may first assume that exactly k symbols are replaced (analogously to the argument given in the proof of [14, Theorem 6.4]). The valence test condition can be checked by the programmed grammar as indicated in the proof of Theorem 3.2. (Things get easier here, since only a bounded number of combinations has to be checked.)

Let $G = (N, T, P, S)$ be a programmed grammar without appearance checking. Let $\mathrm{Lab}(P) = \{p_1, \ldots, p_r\}$ with $(p_i : A_i \to w_i, \sigma_i, \emptyset)$. Let p be a prime number. We give a simulating valence test uniformly k-limited ET0L system $G' = (N \cup T \cup \{S', X, \#\} \cup \mathrm{Lab}(P), T, \mathfrak{P}, S')$ (where $k \geq 2$). X is a special failure symbol. For each p_i and each $q \in \sigma_i$, we have one simulating table:

$$\begin{aligned} P_{i,q} &= \{(A_i \to w_i, p^{k-1})\} \cup \{(p_i \to q, p^{-1})\} \cup \\ &\quad \cup \{(B \to X, 1) \mid B \in \{S', X, \#\} \cup N \cup T \cup (\mathrm{Lab}(P) \setminus \{p_i\})\}. \end{aligned}$$

In addition, we have one initialization and one termination table:

$$\begin{aligned}
\text{init} &= \{(S' \to S\#q^{k-1}, 1) \mid q \in \text{Lab}(P)\} \cup \{(B \to X, 1) \mid B \neq S'\}, \\
\text{exit} &= \{(\# \to \lambda, 1)\} \cup \{(q \to \lambda, 1) \mid q \in \text{Lab}(P)\} \cup \\
&\quad \cup \{(B \to X, 1) \mid B \in N \cup \{S', X\}\}.
\end{aligned}$$

Let $\mathfrak{P} = \{P_{i,q} \mid 1 \leq i \leq r, q \in \sigma_i\} \cup \{\text{init}, \text{exit}\}$.

In the case of propagating systems, we can exploit the closure of programmed languages under derivatives and the easily shown closure of valence test uniformly k-limited EPT0L languages under union in order to use nearly the same construction to prove the claim. □

The proof given above might be slightly simplified but, in the present version, it is also valid in the case of uniformly limited valence ET0L languages, as seen below.

Again, by taking different primes instead of different tables, we can show:

Corollary 3.7 For each $k \geq 2$:

1. $\mathcal{L}(\text{P}, \text{CF}[-\lambda]) = \mathcal{L}(\text{Val}_t, uk\ell\text{E[P]0L})$, and
2. $\mathcal{L}(\text{P}, \text{CF}[-\lambda]) = \mathcal{L}(\text{Val}_t, uk\ell\text{E[P]DT0L})$. □

We return to valence tests in the Sections 6 and 7.

4 The power of valences and valuations

In this section, we start the investigation of valences and valuations in (uniformly) limited ET0L systems. Sometimes, we are only able to state partial results at this point, leaving the answers to our later investigations of closure properties. In particular, we will consider valences and valuations in Lindenmayer systems mostly in later sections.

Firstly, we might interpret valence tests and valences such that we provide our mechanisms with additional counters. In the case of valence tests, the counters must be balanced after each parallel step. In the case of valences, these counters are blind, meaning that a zero-test is only possible in the end, see [16]. By this interpretation, it is intuitively clear that valences and valence tests are incomparable mechanisms.

The simulations of (limited or unlimited) ET0L systems without valences by programmed grammars can be adapted to show that valence ET0L systems can be simulated by programmed grammars with valences. Since valences do not add to the power of programmed grammars *with erasing rules* [2] (with or without appearance checking) [11], we immediately obtain from the theorems of the preceding section:

Corollary 4.1 Let $k \in \mathbb{N}$. Then, we can deduce:

1. $\mathcal{L}(\mathrm{Val}, \mathrm{ET0L}[, \mathbb{Z}]) \subseteq \mathcal{L}(\mathrm{Val}_t, \mathrm{ET0L}[, \mathbb{Z}])$, cf. Theorem 3.2,

2. $\mathcal{L}(\mathrm{Val}, k\ell\mathrm{ET0L}[, \mathbb{Z}]) \subseteq \mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{ET0L}[, \mathbb{Z}])$, cf. Theorem 3.5, and

3. if $k \geq 2$:
   $\mathcal{L}(\mathrm{Val}, uk\ell\mathrm{ET0L}[, \mathbb{Z}]) \subseteq \mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{ET0L}[, \mathbb{Z}])$, cf. Theorem 3.6. □

When do these inclusions turn into equalities? What about propagating systems? (The problem is how to cope with large counters.) What about $u1\ell$ET0L systems? We solve these questions (partially) below.

Theorem 4.2 1. $\mathcal{L}(\mathrm{RE}) = \mathcal{L}(\mathrm{Val}, k\ell\mathrm{ET0L}) = \mathcal{L}(\mathrm{Val}, k\ell\mathrm{ET0L}, \mathbb{Z})$, and

2. $\mathcal{L}(\mathrm{P}, \mathrm{CF} - \lambda, \mathrm{ac}) \subseteq \mathcal{L}(\mathrm{Val}, 1\ell\mathrm{EPT0L}, \mathbb{Z}) \subseteq \mathcal{L}(\mathrm{Val}, 1\ell\mathrm{EPT0L})$.

Proof. According to the last corollary, we only have to show the inclusion from left to right. In the case $k = 1$, this is done as in the proof of Theorem 3.2. X,1) | In order to see the correctness of the construction, observe that any application of P_i to a string which contains non-terminals which does not introduce the failure symbol X will lead to a multiplication of the derivation's valence by p^{-1}. This can only be counterbalanced by applying $(A_i \to w_i, p)$ in the same step, since we are in the case $k = 1$.

In the case of k-limited systems with $k > 1$, the problem arises that more than one rule of the form $A_i \to w_i$ is simulated in one table application, which might be counterbalanced via an incorrect second application of that table, where no left-hand side is present. So, the simulation is more complicated in this case.

Let $G = (N, T, P, S)$ be a programmed grammar with appearance checking. Let $\mathrm{Lab}(P) = \{p_1, \ldots, p_r\}$ with $(p_i : A_i \to w_i, \sigma_i, \varphi_i)$. Let p be a prime number. We give a simulating valence $k\ell$ET0L system $G' = (N \cup T \cup \{X, S', \sigma\} \cup \overline{\mathrm{Lab}}(P), T, \mathfrak{P}, S')$. X is a special failure symbol. $\overline{\mathrm{Lab}}(P)$ consists of unprimed, primed and double-primed versions of rule labels of P. For each label p_i, we have four simulating tables:

[2] Diagram 2.5.1 in [4] claims the inclusion of context-free valence grammars within programmed grammars for the case when erasing rules are forbidden as well, but the given proof works only in the case when erasing rules are allowed.

1. one is responsible for the appearance checking case, i.e.,

$$\begin{aligned} P_{i,ac} &= \{(p_i \to q, 1) \mid q \in \varphi_i\} \cup \\ &\cup \{(B \to B, 1) \mid B \in (N \setminus \{A_i\}) \cup T \cup \{X\}\} \cup \\ &\cup \{(B \to X, 1) \mid B \in \{A_i, S'\} \cup (\overline{\mathrm{Lab}}(P) \setminus \{p_i\})\}\,; \end{aligned}$$

2. three are responsible for the non-appearance checking case, i.e.,

$$\begin{aligned} P_i &= \{(A_i \to w_i\sigma^k, p)\} \cup \{(p_i \to p_i', p^{-1})\} \cup \\ &\cup \{(B \to B, 1) \mid B \in N \cup T \cup \{X\}\} \cup \\ &\cup \{(B \to X, 1) \mid B \in \{S'\} \cup (\overline{\mathrm{Lab}}(P) \setminus \{p_i\})\}\,. \\ P_i' &= \{(\sigma \to \lambda, 1)\} \cup \{(p_i' \to p_i'', 1)\} \cup \\ &\cup \{(B \to B, 1) \mid B \in N \cup T \cup \{X\}\} \cup \\ &\cup \{(B \to X, 1) \mid B \in \{S'\} \cup (\overline{\mathrm{Lab}}(P) \setminus \{p_i'\})\}\,. \\ P_i'' &= \{(\sigma \to X, 1)\} \cup \{(p_i'' \to q, 1) \mid q \in \sigma_i\} \cup \\ &\cup \{(B \to B, 1) \mid B \in N \cup T \cup \{X\}\} \cup \\ &\cup \{(B \to X, 1) \mid B \in \{S'\} \cup (\overline{\mathrm{Lab}}(P) \setminus \{p_i''\})\}\,. \end{aligned}$$

   In the case that no occurrence of A_i is replaced when applying table P_i, this will bring the counter out of balance and, in the case that more than one occurrence of A_i is replaced, only k of the at least $2k$ occurrences of σ's are erased when applying P_i', hence leading to a string containing at least k failure symbols after having applied P_i''.

In addition, we have one initialization and one termination table:

$$\begin{aligned} \mathrm{init} &= \{(S' \to Sq, 1) \mid q \in \mathrm{Lab}(P)\} \cup \{(B \to X, 1) \mid B \neq S'\}, and \\ \mathrm{exit} &= \{(a \to a, 1) \mid a \in T\} \cup \{(q \to \lambda, 1) \mid q \in \mathrm{Lab}(P)\} \cup \\ &\cup \{(B \to X, 1) \mid B \in N \cup \{X, S'\} \cup \overline{\mathrm{Lab}}(P)\} \end{aligned}$$

Hence, $\mathfrak{P} = \{P_{i,ac}, P_i, P_i', P_i'' \mid 1 \leq i \leq r\} \cup \{\mathrm{init}, \mathrm{exit}\}$. □

More precisely, 1-limited valence EPT0L systems characterize programmed grammars with valences and appearance checking (without erasing rules), as investigated in [11], since the incorporation of valences in the proofs showing equivalence of 1ℓEPT0L systems and programmed grammars with unconditional transfer (see [3, 7, 14]) is also valid in presence of valence rules.

Open Problem 2. The case of k-limited systems with $k > 1$ is open if we disallow erasing productions. $\square$

As regards uniformly limited systems, we know by [14, Theorem 6.4] that

$$\bigcup_{k\in\mathbb{N}} \mathcal{L}(uk\ell\mathrm{E[P]T0L}) \subseteq \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda])\,,$$

where the strictness of the inclusion is unknown.

Theorem 4.3 Let $k \in \mathbb{N}$.

- If $k = 1$, we have

$$\begin{aligned} \mathcal{L}(\mathrm{CF}) &\subsetneq \mathcal{L}(\mathrm{Val}, uk\ell\mathrm{E[P][T]0L}, \mathbb{Z}) = \mathcal{L}(\mathrm{Val}, \mathrm{CF}, \mathbb{Z}) \\ &\subsetneq \mathcal{L}(\mathrm{Val}, uk\ell\mathrm{E[P][T]0L}) = \mathcal{L}(\mathrm{Val}, \mathrm{CF})\,. \end{aligned}$$

- Let $k \geq 2$. Then, we find:

  1. $\mathcal{L}(\mathrm{P}, \mathrm{CF}) = \mathcal{L}(\mathrm{Val}, uk\ell\mathrm{ET0L}) = \mathcal{L}(\mathrm{Val}, uk\ell\mathrm{ET0L}, \mathbb{Z})$, and
  2. $\mathcal{L}(\mathrm{P}, \mathrm{CF}{-}\lambda) \subseteq \mathcal{L}(\mathrm{Val}, uk\ell\mathrm{EPT0L}, \mathbb{Z}) \subseteq \mathcal{L}(\mathrm{Val}, uk\ell\mathrm{EPT0L}) \subseteq \mathcal{L}(\mathrm{V}, \mathrm{CF}{-}\lambda)$.

Proof. The assertions concerning case $k = 1$ are obvious through the known relation between context-free valence languages and $u1\ell\mathrm{E[T]0L}$ valence languages. The fact that erasing productions do not add to the power of such systems follows by the Chomsky normal form theorem of context-free valence languages proved in [13], cf. also [11]. The claimed strict inclusions are well-known, see [4].

We show the inclusion $\mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda]) \subseteq \mathcal{L}(\mathrm{Val}, uk\ell\mathrm{E[P]T0L})$ for $k \geq 2$ in the following. To this end, we re-consider the construction given in the proof of Theorem 3.6. We can assume that the programmed grammar we are going to simulate does not contain any rule whose success field contains the rule itself.

Entering a simulating table $P_{i,q}$ without at least one occurrence of a left-hand side A_i present is not possible since, inevitably, a failure symbol will be introduced. So, applying $P_{i,q}$, at least once the rule $A_i \to w_i$ has to be applied. If more than one such rule is applied (either in the case not all occurrences of p_i are replaced, or all p_i and the delimiter symbol # have been erased by a premature application of the terminating table), the counter will be irrevocably out of balance. $\square$

Open Problem 3. For $k > 1$, we conjecture that valence $uk\ell\mathrm{ET0L}$ systems are stronger than valence $uk\ell\mathrm{E0L}$ systems, but we have no proof for this conjecture. Likewise, the relation between valence $uk\ell$EPT0L and valence $uk\ell$E[P]0L systems with varying k is open. □

What about the relation of valuated and valence systems? Here, it is possible to prove a rather general inclusion property.

Theorem 4.4 $\mathcal{L}(\mathrm{Val}', [\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L}[,\mathbb{Z}]) \subseteq \mathcal{L}(\mathrm{Val}, [\mathrm{E}][\mathrm{P}][\mathrm{D}][\mathrm{T}]0\mathrm{L}[,\mathbb{Z}])$. Analogous inclusions are valid for the corresponding families of k-limited and uniformly k-limited systems for any $k \in \mathbb{N}$.

Proof. Let $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ be a valuated ET0L system with $\mathfrak{P} = \{P_1, \ldots, P_n\}$, and let $h : \Sigma^* \to \Delta^*$ denote the natural inclusion morphism. Then, $G' = (\Sigma, \Delta, \mathfrak{P}', \omega)$ with $\mathfrak{P}' = \{P_1', \ldots, P_n'\}$, where $(a \to w, v) \in P_i'$ iff $a \to w \in P_i$ and $v = \varphi(h(w))/\varphi(h(a))$. □

We will return to the question about strictness of the inclusions of the last theorem later.

Finally, let us discuss $uk\ell$ET0L systems a bit in detail. The proof of [32, Theorem 2.2] carries over to valence and valuated such systems (and grammars). It is quite clear that valuated context-free grammars are strictly more powerful than context-free grammars. The following example shows that valuated context-free grammars are strictly less powerful than valence context-free grammars. Namely, consider the valence grammar $(\{S, B\}, \{a, b\}, \{(S \to aSa, 1), (S \to aBa, 1), (S \to b, 0), (B \to bB, -1), (B \to b, -1)\}, S)$ which generates $\{a^n b^{n+1} a^n \mid n \geq 0\} =: L$. Assume L can be generated by a context-free valuated grammar with valuation morphism φ. Since $b \in L$, $\varphi(b) = 1$. Also, since $abba \in L$, $\varphi(a) = 1$. Therefore, L must be context-free, which is obviously not the case.

Furthermore, it is known by [4, Theorem 2.1.4] that context-free valence grammars are as powerful as context-free unordered vector grammars, which form a strict subclass of programmed languages by Theorem 2.1.3 in combination with Theorems 2.1.2 and 1.2.2 of [4]. This leads (together with Theorem 3.6) to the following result.

Corollary 4.5

$$\begin{array}{lllll}
\mathcal{L}(\mathrm{CF}) & = & \mathcal{L}(u1\ell\mathrm{E}[\mathrm{P}]\mathrm{T0L}) & = & \mathcal{L}(\mathrm{Val}_t, u1\ell\mathrm{E}[\mathrm{P}]\mathrm{T0L}) \subsetneq \\
 & \subsetneq & \mathcal{L}(\mathrm{Val}', \mathrm{CF}) & = & \mathcal{L}(\mathrm{Val}', u1\ell\mathrm{E}[\mathrm{P}]\mathrm{T0L}) \subsetneq \\
 & \subsetneq & \mathcal{L}(\mathrm{Val}, \mathrm{CF}) & = & \mathcal{L}(\mathrm{Val}, u1\ell\mathrm{E}[\mathrm{P}]\mathrm{T0L}) \subsetneq \\
 & \subsetneq & \mathcal{L}(\mathrm{P}, \mathrm{CF}[-\lambda]) & & \square
\end{array}$$

In particular, observe that, in this case, valence tests is the weakest form of regulation while, generally speaking, it is the strongest form of regulation.

It is of natural interest to compare valence systems with their non-valence counterparts. When we do not admit more than one table, we have the following result.

Theorem 4.6 Let $k \in \mathbb{N}$. Then,

$$\begin{aligned}\mathcal{L}(uk\ell\text{E[P]0L}) &= \mathcal{L}(\text{Val}_t, uk\ell\text{E[P]0L}) \subsetneq \\ &\subsetneq \mathcal{L}(\text{Val}', uk\ell\text{E[P]0L}) \subsetneq \mathcal{L}(\text{Val}, uk\ell\text{E[P]0L}).\end{aligned}$$

Moreover, for $X = \text{Val}$ and $X = \text{Val}'$ we have the inclusion

$$\mathcal{L}(X, \text{CF}) \subseteq \mathcal{L}(X, uk\ell\text{E[P]0L}).$$

Proof. $uk\ell$E0L systems characterize the k-context-free languages introduced by K. Salomaa [25], as shown in [32]. K. Salomaa has proved that

$$L(k) = \{a_1^n \cdots a_{2k+1}^n \mid n \geq 1\} \subset \{a_1, \ldots, a_{2k+1}\}^*$$

cannot be generated by a $uk\ell$E0L system. Taking k different primes $p_1, \ldots, p_k$, the valuated system

$$G(k) = (\{A_1, \ldots, A_{k+1}, a_1, \ldots, a_{2k+1}\}, \{a_1, \ldots a_{2k+1}\}, P, A_1 \cdots A_{k+1}, \varphi)$$

generates $L(k)$ for any $k \geq 1$, where

$$\begin{aligned}P &= \{A_i \to a_{2i-1}A_i a_{2i}, A_i \to a_{2i-1}a_{2i} \mid 1 \leq i \leq k\} \cup \\ &\cup \{A_{k+1} \to A_{k+1}a_{2k+1}, A_{k+1} \to a_{2k+1}\} \cup \\ &\cup \{a_i \to a_i \mid 1 \leq i \leq 2k+1\}; \\ \varphi(a_{2i-1}) &= 1 \text{ and} \\ \varphi(a_{2i}) &= p_i^{-1}, \text{ for } 1 \leq i \leq k, \quad \text{and} \\ \varphi(a_{2k+1}) &= p_1 \cdots p_k\,.\end{aligned}$$

The class of languages generated by k-context-free languages (which equals $\mathcal{L}(uk\ell\text{E[P]0L})$ due to [32]) is closed under [non-erasing] gsm mappings, see Theorem 7.2 below. Therefore,

$$L'(k) = \{(a^n b)^{2k+1} \mid n \geq 0\} \notin \mathcal{L}(uk\ell\text{E[P]0L}),$$

since, otherwise, $L(k)$ would be a k-context-free language, contradicting Salomaa's result. Assume that (for some $k \geq 1$) $L'(k)$ can be generated by some valuated $uk\ell$E0L system. Then, since $b^{2k+1} \in L'(k)$, $\varphi(b) = 1$, and with

$(ab)^{2k+1} \in L'(k)$ it follows that $\varphi(a) = 1$. So, $L'(k)$ would be in $\mathcal{L}(uk\ell\mathrm{E[P]0L})$, contradicting our previous observations.

On the other hand, by a construction similar to grammar $G(k)$ in the first part of this proof, for any $k' \geq 1$,

$$L'(k) \in \mathcal{L}(\mathrm{Val}, \mathrm{CF}) \subseteq \mathcal{L}(\mathrm{Val}, uk'\ell\mathrm{E[P]0L}).$$

Finally, we come to the "moreover-part". The claimed inclusion is quite easy, since it is possible to blow up a context-free rule set by adding rules of the form $a \to a$ which are for free. □

Open Problem 4. Is the inclusion in the "moreover-part" of the preceding theorem strict in the case of $k \geq 2$? The question seems to be hard, since the counterexamples found for valence context-free grammars and k-context-free grammars are the same. Of course, one can show a trade-off between the admitted parallelism in k-context-free grammars (which is counted by the parameter k) and the number of distinct prime numbers of the values occurring in the grammar. □

5 The power of table valences

Table valences seem to be the mechanism that is nearest to the model without valences, because no valences are attached to rules. Indeed, we can generalize some methods that are not extendible to ET0L systems with valence rules or valuations. For instance, we can make use of the *combinatorially complete version* to prove decidability results. As regards the generative capacity, we can show:

Theorem 5.1 $\mathcal{L}(\mathrm{ET0L}) \subsetneq \mathcal{L}(\mathrm{E(Val)T0L}) \subseteq \mathcal{L}(\mathrm{Val}, \mathrm{ET0L})$.

Remark 5.2 The strictness of the second inclusion will be shown at the end of this section.

Proof. To show the properness of the first inclusion we consider the ET0L system with table valences $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ where

$$\Sigma = \{a, b, A, B, C_1, C_2, X\},$$

$$\Delta = \{a, b\},$$

$$\mathfrak{P} = \{P_1, P_1', P_2, P_3\}$$

$$P_1 = \{B \to B, C_1 \to BC_1\} \cup \{Y \to X \mid Y \in \{a, b, A, C_2, X\}\},$$
$$P_1' = P_1,$$
$$P_2 = \{B \to AB, A \to A, C_1 \to C_2, C_2 \to C_2\} \cup$$
$$\cup \{Y \to X \mid Y \in \{a, b, X\}\},$$
$$P_3 = \{A \to a, B \to b, C_1 \to \lambda, C_2 \to \lambda\} \cup$$
$$\cup \{Y \to X \mid Y \in \{a, b, X\}\},$$

$\omega = C_1$, and

$\varphi(P_1) = 2$, $\varphi(P_1') = 1$, $\varphi(P_2) = 1/2$, $\varphi(P_3) = 1$.

In a derivation using G, we first apply the tables P_1 and P_1', thereby producing every pair of the form $(B^n C_1, 2^m), n \geq m \geq 0$. Next, the table P_2 is used k times, transforming $(B^n C_1, 2^m)$ into $((A^k B)^n C_2, 2^{m-k})$ if $k > 0$. Once P_2 is used, P_1 and P_1' can never be taken again without producing a word containing the trap symbol X. Finally, by applying P_3 one time, we obtain the pair $((a^k b)^n, 2^{m-k})$. After this step, no further derivation is possible without producing a trap symbol. The derivation is valid iff $m = k$. Hence, G generates the language $L(G) = \{(a^m b)^n \mid n \geq m \geq 0\}$ which is not an ET0L language, see [24, Cor. V.2.2].

Now let $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ be an ET0L system with table valences, where $\mathfrak{P} = \{P_1, \ldots, P_k\}$. We construct the valence ET0L system $G' = (\Sigma', \Delta', \mathfrak{P}', \omega')$ with

$$\begin{array}{rcl} \Sigma' & = & \Sigma \cup \{S\},\ S \notin \Sigma, \\ \Delta' & = & \Delta \cup \{S\}, \\ \omega' & = & \omega S, \text{ and} \\ \mathfrak{P}' & = & \{P_1', \ldots, P_k'\}, \text{ where} \\ P_i' & = & \{(a \to w, 1) \mid a \to w \in P_i\} \cup \{(S \to S, \varphi(P_i)\},\ \text{for } 1 \leq i \leq k. \end{array}$$

Obviously, $L(G') = L(G)\{S\}$. According to Theorem 7.5 below, $L(G)$ can be generated by a valence ET0L system, too. □

Theorem 5.3 $\mathcal{L}(\mathrm{E(Val)T0L}) = \mathcal{L}(\mathrm{EP(Val)T0L})$.

Proof. The same construction as in the proof of $\mathcal{L}(\mathrm{ET0L}) = \mathcal{L}(\mathrm{EPT0L})$ can be applied, see [24, Theorem V.1.6]. □

Similar to plain ET0L systems we can assign to a table valence ET0L system G a table valence EDT0L system $\overline{G}$, called the *combinatorially complete version* of G. More specifically, for $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ with $\mathfrak{P} = \{P_1, \ldots, P_k\}$, we define

$\overline{G} = (\Sigma, \Delta, \overline{\mathfrak{P}}, \omega, \overline{\varphi})$ with $\overline{\mathfrak{P}} = \bigcup_{i=1}^{k} \mathfrak{P}_i$ where $\mathfrak{P}_i$ consists of all deterministic tables (homomorphisms) that are subsets of P_i, and $\overline{\varphi}(P) = \varphi(P_i)$, for all $P \in \mathfrak{P}_i$, $1 \leq i \leq k$. As in [24, Lemma V.2.8], we can show:

Lemma 5.4 If $\overline{G}$ is the combinatorially complete version of G, then $L(\overline{G}) \subseteq L(G)$. Moreover, $L(\overline{G})$ is empty (finite) iff $L(G)$ is empty (finite).

Proof. The inclusion $L(\overline{G}) \subseteq L(G)$ follows directly from the definition.

Let $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ be a given table valence ET0L system with the set of tables $\mathfrak{P} = \{P_1, \ldots, P_k\}$. Consider a derivation of a pair $(w_n, r), w_n \in \Delta^*$, from a pair $(w_0, s), w_0 \in \Sigma^*$, using a sequence of tables $P_{i_1}, P_{i_2}, \ldots P_{i_n}$.

We claim that a pair $(w_n', r), w_n' \in \Delta^*, |w_n'| \geq |w_n|$ can be derived from (w_0, s) using the sequence $P_1' \in \mathfrak{P}_{i_1}$, $P_{i_2}, \ldots, P_{i_n}$. Indeed, for $a \in \Sigma$, let $\alpha_{i_1, i_2, \ldots, i_n}(a)$ denote a word of maximal length in Δ^* derivable from a using the sequence $P_{i_1}, P_{i_2}, \ldots, P_{i_n}$. Let $\beta_{i_1, i_2, \ldots, i_n}(a)$ be the string obtained after the first parallel step of this derivation. We can now choose $P_1' = \{a \to \beta_{i_1, i_2, \ldots, i_n}(a) \mid a \in \Sigma\}$. Using the sequence $P_1', P_{i_2}, \ldots, P_{i_n}$, we can generate the word $w_n' = h(w_0)$, where $h : \Sigma^* \to \Delta^*$ is the homomorphism with $h(a) = \alpha_{i_1, i_2, \ldots, i_n}(w_0)$, for $a \in \Sigma$. Obviously, $|w_n'| \geq |w_n|$. Since P_1' has the same valence as P_{i_1}, we can also derive (w_n', r) from (w_0, s).

By an iterated application of the above construction, we can show the existence of a sequence $P_1', P_2', \ldots, P_n'$, $P_j' \in \mathfrak{P}_{i_j}, 1 \leq j \leq n$, that generates, starting from (w_0, s), a pair $(\overline{w_n}, r)$ with $\overline{w_n} \in \Delta^*$ and $|\overline{w_n}| \geq |w_n|$. In particular, we have shown that for all $w \in L(G)$, there is a word $\overline{w} \in L(\overline{G})$ with $|\overline{w}| \geq |w|$. Hence, $L(\overline{G})$ is empty (finite) iff $L(G)$ is empty (finite). $\square$

The benefit of the last two results is that we can often restrict ourselves to EPDT0L systems with table valences. For deterministic systems, the outcome of a derivation can be described by the sequence of applied tables or, equivalently, by a word over the alphabet of the (labels of the) tables. The set of valid derivations (yielding a terminal word and valence 1) can, in the case of EDT0L systems with table valences, be described by a deterministic finite valence automaton. More specifically, for an EDT0L system with table valences $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$, we define the valence automaton $\mathcal{A} = (Z, X, z_0, \delta, F)$ with $Z = \mathcal{P}(\Sigma)$, $X = \mathfrak{P}$, $z_0 = \mathrm{Alph}(\omega)$, $F = \mathcal{P}(\Delta)$, and $\delta(\mathrm{Alph}(w), P) = (\mathrm{Alph}(P(w)), \varphi(P))$, for $P \in \mathfrak{P}$, $w \in \Sigma^*$. Note that δ is well-defined, as $\mathrm{Alph}(w) = \mathrm{Alph}(v)$ implies $\mathrm{Alph}(P(w)) = \mathrm{Alph}(P(v))$, for all $v, w \in \Sigma^*$ and any homomorphism $P : \Sigma^* \to \Sigma^*$. As usual, δ can be extended to a mapping $\delta^* : \mathfrak{P}^* \times Z \to Z \times \mathbb{Q}_+$. Observe that words over $\mathfrak{P}$ are seen as input words for $\mathcal{A}$ and as homomorphisms on Σ^*. It is easily shown by induction that

$$\delta^*(z_0, P_1 \cdots P_n) = (z, r), \text{ where } z = \mathrm{Alph}(P_1 \cdots P_n(\omega)), r = \varphi(P_1) \cdots \varphi(P_n).$$

If, moreover, G is propagating then all morphisms $\alpha = P_1 \cdots P_n$ are non-erasing. In this case, a pair (Γ, α), $\Gamma \subseteq \Sigma$, is called *stagnating* if $|\alpha(a)| = 1$, for all $a \in \Gamma$. Otherwise, it is called *increasing.* For all w with $\mathrm{Alph}(w) = \Gamma$, $|\alpha(w)| = |w|$, if (Γ, α) is stagnating, and $|\alpha(w)| > |w|$, if (Γ, α) is increasing.

Theorem 5.5 The membership, the emptiness, and the finiteness problems are decidable for a given ET0L system with table valences.

Proof. By Theorem 5.3, it suffices to prove decidability of the questions for EPT0L systems with table valences. As regards the membership problem, an EPT0L system with table valences can easily be simulated by a monotone valence grammar. As shown in [22, Theorem 5], for any monotone valence grammar, an equivalent monotone grammar can be constructed.

As regards the emptiness and finiteness problems, we can, by Lemma 5.4, restrict ourselves to the propagating and deterministic case. Let $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ be an EPDT0L system with table valences. Let $\mathcal{A}$ be the deterministic finite valence automaton associated with G. To decide whether $L(G)$ is empty, we have just to decide whether $L(\mathcal{A})$ is empty. To decide finiteness of $L(G)$, we extend $\mathcal{A}$ to a valence transducer. In state $z \subseteq \Sigma$ and with input $P \in \mathfrak{P}$, the transducer outputs the empty word, if (z, P) is stagnating, and a fixed symbol $a \in \Sigma$, if (z, P) is increasing. A word a^n belongs to the image of the transduction iff there is a valid derivation with exactly n increasing derivation steps. Hence, $L(G)$ is finite iff the image of the valence transduction is finite. We have effectively reduced the emptiness and finiteness problems to the analogous problems for finite valence automata, which are decidable. $\square$

Remark 5.6 As regards computational complexity, we just remark that membership and finiteness are PSPACE-complete for EPT0L systems with valences, valuations or table valences, since the necessary counter operations can be done in polynomial space, while the hardness follows by older results contained in [19] for EPT0L systems.

Finally, we are going to show that table valences are strictly weaker than valences attached to rules. We do this by showing that the language $L_0 = \{a^{2^{2^n}} \mid n \geq 0\}$ cannot be generated by an ET0L system with table valences.

To this end, we need some results on homomorphisms. A homomorphism from a monoid M to M is called a homomorphism *on* M. The set of these homomorphisms is denoted by $Hom(M)$; it is a monoid, using the usual composition of functions as operation. (For two functions $f, g : M \to M$, their composition is the function $fg : M \to M$ where $fg(x) = g(f(x))$, for $x \in M$.) We shall now associate homomorphisms on Σ^* with homomorphisms on $\mathbb{N}^r$, where $r = |\Sigma|$.

Let $\Psi : \Sigma^* \to \mathbb{N}^r$ be the Parikh mapping. For a homomorphism $\alpha \in Hom(\Sigma^*)$, we define the mapping $h_\alpha : \mathbb{N}^r \to \mathbb{N}^r$ by $h_\alpha(\Psi(w)) := \Psi(\alpha(w))$. It is easy to verify that h_α is well-defined and a homomorphism. Note that the mapping $H : Hom(\Sigma^*) \to Hom(\mathbb{N}^r)$ with $H(\alpha) = h_\alpha$ is also a homomorphism, i.e., $h_{\alpha\beta} = h_\alpha h_\beta$.

On $\mathbb{N}^r$, let $\leq$ be the natural partial order, defined by the component-wise $\leq$-relation. We write "$a < b$" as an abbreviation for "$a \leq b$ and $a \neq b$".

Lemma 5.7 Any homomorphism $h \in Hom(\mathbb{N}^r)$ is monotonous, i.e., $h(\vec{x}) \leq h(\vec{y})$, for all $\vec{x} < \vec{y}$. Moreover, if h is non-erasing, i.e., $h(\vec{x}) = \vec{0} \iff \vec{x} = \vec{0}$, then $h(\vec{x}) < h(\vec{y})$, for all $\vec{x} < \vec{y}$.

Proof. $\vec{x} < \vec{y}$ holds iff there is a vector $\vec{a} \in \mathbb{N}^r \setminus \{\vec{0}\}$ such that $\vec{x} + \vec{a} = \vec{y}$. As h is a homomorphism, we obtain $h(\vec{x}) \leq h(\vec{x}) + h(\vec{a}) = h(\vec{x} + \vec{a}) = h(\vec{y})$. The equality $h(\vec{x}) = h(\vec{y})$ holds iff $h(\vec{a}) = \vec{0}$. □

Lemma 5.8 Let H be a finite set of homomorphisms on $\mathbb{N}^r$. There are constants $m, p \in \mathbb{N}$ such that, for all $\vec{x} \in \mathbb{N}^r$, $n \geq m$ and $h \in H$, we have: $h^n(\vec{x}) \leq h^{n+p}(\vec{x})$.

Proof. We first consider the case that H consists of a single homomorphism h. Let $\vec{e}_i$, $1 \leq i \leq r$, be the i-th unit vector in $\mathbb{N}^r$. Due to Dickson's Lemma [5], there are numbers m_i and n_i such that $m_i < n_i$ and $h^{m_i}(\vec{e}_i) \leq h^{n_i}(\vec{e}_i)$. Set $p_i = n_i - m_i > 0$. For all $n \geq m_i$ and $\ell \in \mathbb{N}$, we have

$$h^{n+\ell p_i}(\vec{e}_i) = h^{n+\ell p_i - m_i}(h^{m_i}(\vec{e}_i)) \leq h^{n+\ell p_i - m_i}(h^{n_i}(\vec{e}_i)) = h^{n+(\ell+1)p_i}(\vec{e}_i)$$

according to the preceding lemma. By induction, we get, for all $\ell \in \mathbb{N}$,

$$h^n(\vec{e}_i) \leq h^{n+\ell p_i}(\vec{e}_i). \tag{3}$$

Now, choose $m_h = \max\{m_i \mid 1 \leq i \leq r\}$ and $p_h = p_1 \cdots p_r$. Set $\ell_i = p_h/p_i$, $1 \leq i \leq r$ (note that $\ell_i \in \mathbb{N}$). For all $n \geq m$, $j \in \mathbb{N}$ and $\vec{x} \in \mathbb{N}^r$, where $\vec{x} = (x_1, \ldots, x_r)$, we can deduce, using Eq. (3),

$$\begin{aligned} h^n(\vec{x}) &= \sum_{i=1}^{r} h^n(x_i\vec{e}_i) = \sum_{i=1}^{r} x_i h^n(\vec{e}_i) \\ &\leq \sum_{i=1}^{r} x_i h^{n+j\ell_i p_i}(\vec{e}_i) = \sum_{i=1}^{r} h^{n+jp_h}(x_i\vec{e}_i) = h^{n+jp_h}(\vec{x})\,. \end{aligned}$$

If H consists of a finite number of morphisms h with constants m_h and p_h, we obtain the claimed constants as $m = \max\{m_h \mid h \in H\}$ and $p = \prod_{h\in H} p_h$, since $h^n(\vec{x}) \leq h^{n+jp_h}(\vec{x})$ for $j = p/p_h$. □

Lemma 5.9 There is no EPDT0L system with table valences G such that $L(G)$ is an infinite subset of $L_0 = \{a^{2^{2^n}} \mid n \geq 0\}$.

Proof. Assume, for contradiction, the existence of an EPDT0L system with table valences G such that $L(G)$ is an infinite subset of L_0. Let $\mathcal{A} = (Z, X, z_0, \delta, F)$ be the deterministic finite valence automaton associated with G. A *cycle* (with respect to $\mathcal{A}$) is a triple $(z, \beta, q) \in Z \times X^+ \times \mathbb{Q}_+$ with $\delta^*(z, \beta) = (z, q)$. A cycle (z, β, q) is called *stagnating* (*increasing*) if (z, β) is stagnating (increasing). We say that a cycle $\zeta = (z, \beta, q)$ *appears in* $\alpha \in X^*$ iff there is a decomposition $\alpha = \alpha_1 \beta \alpha_2$ with $\delta^*(z_0, \alpha_1) \in \{z\} \times \mathbb{Q}_+$. The number of appearances of ζ in α is denoted by $n(\zeta, \alpha)$. As $L(G)$ is infinite, there must be an increasing cycle $\hat{\zeta} = (\hat{z}, \hat{\beta}, \hat{q})$ such that $\sup\{n(\hat{\zeta}, \alpha) \mid \alpha \in L(\mathcal{A})\} = \infty$.

It can be shown (compare with [10]) that there are

1. cycles $(z_1, \beta_1, q_1), \ldots, (z_k, \beta_k, q_k)$ with $\prod_{i=1}^k q_i = 1$, and $z_t = \hat{z}, \beta_t = \hat{\beta}^\rho, q_t = \hat{q}^\rho$ for some $1 \leq r \leq k$ and some $\rho \geq 1$, and
2. a word word $\alpha \in L(A)$ such that $\alpha = \alpha_1 \cdots \alpha_k \alpha_{k+1}$, where $\delta^*(z_{i-1}, \alpha_i) = (z_i, s_i)$ for some $s_i \in \mathbb{Q}_+$, with $1 \leq i \leq k$.

Note that all words (which are, at the same time, homomorphisms on Σ^*)

$$\gamma_j := \alpha_1 \beta_1^j \alpha_2 \cdots \alpha_k \beta_k^j \alpha_{k+1},\ j \in \mathbb{N}$$

are in $L(\mathcal{A})$. The Parikh vector of the word $\gamma_j(\omega)$ is obtained as

$$\Psi(\gamma_j(\omega)) = h_{\gamma_j}(\Psi(\omega)) = h_{\alpha_1} h_{\beta_1}^j \cdots h_{\alpha_k} h_{\beta_k}^j h_{\alpha_{k+1}}(\Psi(\omega)).$$

According to Lemma 5.8, there are constants m, p such that $h_{\beta_i}^j(\vec{x}) \leq h_{\beta_i}^{j+p}(\vec{x})$, for all $j \geq m$, $1 \leq i \leq k$ and $\vec{x} \in \mathbb{N}^r$. Taking into account Lemma 5.7, for $n \geq 0$, we can conclude that $\Psi(\gamma_{m+np}(\omega)) \leq \Psi(\gamma_{m+(n+1)p}(\omega))$. Since at least one of the cycles (z_i, β_i) is increasing, we get even $\Psi(\gamma_{m+np}(\omega)) < \Psi(\gamma_{m+(n+1)p}(\omega))$ due to Lemma 5.7. As $\gamma_{m+np}(\omega)$ is in $L(G) \subseteq L_0$, this implies $|\gamma_{m+np}(\omega)| \geq 2^{2^n}$, for all $n \geq 0$.

On the other hand, there is a constant c (dependent on the EDT0L system G), such that $|\gamma_{m+np}(\omega)| \leq c^{|\gamma_{m+np}|}$, meaning that $|\gamma_{m+np}(\omega)| < 2^{2^n}$, for sufficiently large n. □

Theorem 5.10 $\mathcal{L}(\mathrm{E(Val)T0L}) \subsetneq \mathcal{L}(\mathrm{Val}, \mathrm{ET0L})$.

Proof. The language $L_0 = \{a^{2^{2^n}} \mid n \geq 0\}$ cannot be generated by a table valence system, since otherwise the combinatorially complete version would generate an infinite subset of L_0. A valence ET0L system over $\mathbb{Z}$ generating L_0 is $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$ with $\Sigma = \{A, B, C, X, a\}$, $\Delta = \{a\}$, $\omega = AC$ and $\mathfrak{P} = \{P_1, P_2, P_3, P_4\}$ with

$$\begin{aligned}
P_1 &= \{(A \to AA, 1), (B \to X, 0), (C \to C, 0), (X \to X, 0), (a \to X, 0)\},\\
P_2 &= \{(A \to \lambda, 0), (B \to X, 0), (C \to BC, 1), (X \to X, 0), (a \to X, 0)\},\\
P_3 &= \{(A \to X, 0), (B \to BB, 0), (C \to C, -1), (X \to X, 0), (a \to X, 0)\},\\
P_4 &= \{(A \to X, 1), (B \to a, 0), (C \to \lambda, 0), (X \to X, 0), (a \to X, 0)\}.
\end{aligned}$$

A word over $\{a\}$ can be generated by applying, in this sequence, n times P_1, yielding $(A^{2^n}C, 2^n - 1)$, once P_2, yielding $(BC, 2^n)$, k times P_3, yielding $(B^{2^k}C, 2^n - k)$, and once P_4, yielding $(a^{2^k}, 2^n - k)$. To obtain a neutral valence, $k = 2^n$ must hold. □

Remark 5.11 In a similar manner, one can show that for each fixed k, $L_k = \{ \underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{k \text{ times}}{}^{n} \mid n > 0\}$ lies in $\mathcal{L}(\mathrm{Val}, \mathrm{ET0L})$.

Open Problem 5. Among the open questions concerning table valence systems, we only mention that we did not treat (uniformly) limited systems here, and we did not consider Lindenmayer systems with just one table, either. □

6 Normal forms

6.1 ET0L versus T0L

We shall first show the surprising fact that the introduction of terminal symbols does (nearly) not increase the power of valence/valuated L systems. To the contrary, in the case of (uniformly limited or limited) valence test ET0L systems, we get strict inclusions.

Theorem 6.1 Let Δ be an alphabet, S be a symbol not in Δ, and $L \subseteq \Delta^*$ be a language. Let $k \in \mathbb{N}$. The following relations are true:

1. $L \in \mathcal{L}(\mathrm{Val}', \mathrm{E[T]0L}) \iff L \in \mathcal{L}(\mathrm{Val}', \mathrm{[T]0L})$,
2. $L \in \mathcal{L}(\mathrm{Val}, \mathrm{E[T]0L}) \iff L \cup \{S\} \in \mathcal{L}(\mathrm{Val}, \mathrm{[T]0L})$,
3. $\mathcal{L}(\mathrm{Val}_t, \mathrm{ET0L})$ is a strict superset of $\mathcal{L}(\mathrm{Val}_t, \mathrm{T0L})$.
4. $L \in \mathcal{L}(\mathrm{Val}', (u)k\ell\mathrm{E[T]0L}) \iff L \in \mathcal{L}(\mathrm{Val}', (u)k\ell\mathrm{[T]0L})$,

5. $L \in \mathcal{L}(\text{Val}, (u)k\ell\text{E[T]0L}) \iff L \cup \{S\} \in \mathcal{L}(\text{Val}, (u)k\ell\text{[T]0L})$,

6. $\mathcal{L}(\text{Val}_t, (u)k\ell\text{E[T]0L})$ is a strict superset of $\mathcal{L}(\text{Val}_t, (u)k\ell\text{[T]0L})$.

Proof.
The directions "$\Leftarrow$" are trivially true.

1. Let $G = (\Sigma, \Delta, \mathfrak{P}, \omega, \varphi)$ be a valuated ET0L system such that Δ is a proper subset of Σ. We choose a prime number $p \notin \text{Primes}(G)$. The valuated T0L system system $G' = (\Sigma, \Sigma, \mathfrak{P}, \omega, \varphi')$ with $\varphi'(a) = \varphi(a)$ for $a \in \Delta$, and $\varphi'(a) = p$ for $a \in \Sigma \setminus \Delta$, obviously generates the same language as G.

   Note that in this proof the number of tables was not changed. The assertion has thus been shown for valuated E0L systems as well.

2. Now let $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$ be a valence ET0L system such that Δ is a proper subset of Σ. Let $p \notin \text{Primes}(G)$ be a prime number. We now define the valuation mapping $\varphi : \Sigma \to \mathbb{Q}_+$ by $\varphi(a) = p$, if $a \in \Sigma \setminus \Delta$, and $\varphi(a) = 1$, if $a \in \Delta$. As above, φ can be extended to words over Σ.

   The valence T0L system system $G' = (\Sigma \cup \{S\}, \Sigma \cup \{S\}, \mathfrak{P}', S)$ is constructed in the following way. For all $P \in \mathfrak{P}$, we define the table $P' \in \mathfrak{P}'$ as

   $$P' = \{(S \to \omega, \varphi(\omega))\} \cup \{(a \to w, r \cdot \varphi(w)/\varphi(a)) \mid (a \to w, r) \in P\}.$$

   It can easily be shown by induction that a pair (w, r) can be derived in G iff $(w, r \cdot \varphi(w))$ is generated in G'. Hence, a word in $L(G')$ cannot contain a symbol from $\Sigma \setminus \Delta$. Since the axiom S is trivially contained in the language of a T0L system, we get $L(G') = \{S\} \cup L(G)$. Again, the number of tables was not changed. Thus the assertion holds for valence E0L systems as well.

3. Intuitively, the proof given in the case of valence grammars does not work in this case, since we cannot collect information on nonterminal occurrences within the valence number.

   Consider the language $L = \{a^n b^n a^{2n+1} \mid n \geq 1\}$. Suppose there is a valence test T0L system $G = (\{a, b\}, \mathfrak{P}, \omega)$ such that $L = L(G)$. Define h as $h = \max\{|\omega|, \max\{|\alpha| \mid (c \to \alpha, r) \in \bigcup \mathfrak{P},$ for some $c \in \{a, b\}, r \in \mathbb{Q}_+\}\}$. For some $n > h$, the word $w = a^n b^n a^{2n+1}$ is directly derivable from a word $v = a^m b^m a^{2m+1}$ such that $m < n$. We can assume that $m \geq 2$.

If some appearance of a was, in this derivation step, replaced by a word α containing a b we could find a derivation from v where the first a is replaced by α and would get a shape-breaking word. Hence, all appearances of a in v are substituted by words in a^*. In a similar way, it is shown that any b is replaced by a word in b^*. Moreover, not all appearances of a can be replaced by the same word a^r, $r > 1$. Otherwise, the generated word would have the form $a^{mr}b^x a^{2mr+r}$.

Hence, there are two different numbers q_1, q_2 such that one of the first m appearances of a in v is replaced by a^{q_1} and one of the last $2m+1$ appearances of a in v is replaced by a^{q_2}. By exchanging these substitutions, one can derive a word that is not in L. We have thus shown that $L \notin \mathcal{L}(\mathrm{Val}_t, [\mathrm{T}]0\mathrm{L})$.

On the other hand, it is easy to see that L is an E0L language.

4. The first part of this proof is valid here, too.
5. The second part of this proof is valid here, too.
6. The third part of this proof is valid here, too. □

6.2 The degree of synchronization

As regards the number of tables (the so-called degree of synchronization) in ET0L systems (with valuations), we get the following result:

Theorem 6.2 1. If $L \in \mathcal{L}(\mathrm{Val}', \mathrm{ET0L})$, then L can be generated by a valuated ET0L system having two tables.

2. If $L \in \mathcal{L}(\mathrm{Val}, \mathrm{ET0L})$, then L can be generated by a valence ET0L system having two tables.
3. If $L \in \mathcal{L}(\mathrm{Val}_t, \mathrm{ET0L})$, then L can be generated by a valence test ET0L system having two tables.

Corresponding results hold for $k\ell$ET0L systems, too.

Proof. The proof of the same fact on ET0L systems, given in [24, Theorem V.1.3], or for $k\ell$ET0L systems, given in [31], readily transfers to our case. □

Open Problem 6. Actually, we do not have a proof that one table does not suffice for those systems. Combinatorial properties as for E0L systems in [24] are still to be discovered. For (limited) ET0L systems with valence tests, the result can be sharpened according to the next theorems. □

Open Problem 7. To our knowledge, the problem of synchronization degree is still open for uniformly limited ET0L systems. The difficulty is to control the sequence of table applications, so the problem is similar to the question of whether $uk\ell$ET0L languages coincide with the programmed languages without appearance checking or not. □

In the cases of limited systems with valence tests, the results from above can be sharpened, as already indicated in Section 3 when discussing simulations of programmed grammars.

Theorem 6.3 For $k \in \mathbb{N}$, $\mathcal{L}(\mathrm{Val}_t, (u)k\ell\mathrm{E[P]0L}) = \mathcal{L}(\mathrm{Val}_t, (u)k\ell\mathrm{E[P]T0L})$.

Proof. First, we consider uniform limitations. Let $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$ be a uniformly k-limited EPT0L system with the set of tables $\mathfrak{P} = \{P_1, \ldots, P_m\}$. Choose m different primes $p_1, \ldots, p_m$ not in Primes(G).

We now construct the uniformly k-limited EP0L system G' as follows: $G' = (\Sigma \cup \Sigma' \cup \{S\}, \Delta, P, S)$ where Σ' is a copy of Σ, and $S \notin \Sigma \cup \Sigma'$. For a word $w \in \Sigma^+$, let $D_1(w)$ denote the set of all words obtained from w by changing exactly one letter in w to its copy in Σ'. As usual, for a language $L \subseteq \Sigma^*$, we set $D_1(L) = \{w' \mid w' \in D_1(w) \text{ for some } w \in L\}$. With this notation, we give the set of valence rules P:

$$\begin{aligned}
P \;=\; & \{(S \to \omega', 1) \mid \omega' \in D_1(\omega)\} \\
& \cup \bigcup_{i=1}^{m} \{(a \to w, rp_i^{-1}) \mid (a \to w, r) \in P_i\} \\
& \cup \bigcup_{i=1}^{m} \{(a' \to w', rp_i^{j}) \mid (a \to w, r) \in P_i, w' \in D_1(w), 0 \leq j \leq k-1\} \\
& \cup \bigcup_{i=1}^{m} \{(a' \to w, rp_i^{j}) \mid (a \to w, r) \in P_i, 0 \leq j \leq k-1\}.
\end{aligned}$$

Firstly, note that all derivable words in G' contain at most one symbol from Σ'. Moreover, no further derivation is possible from a word in Σ^* since, otherwise, the valence product would contain a negative power of some p_i. Hence, a derivation in G' produces a sequence of words in $D_1(\Sigma^*)$ and finishes with at most one word over Σ. It is easily shown by induction that a word $w' \in D_1(\Sigma^*)$ can be derived in G' iff the (unique) word w with $w' \in D_1(w)$ is derivable in G. By using the last line of the definition of P in the last derivation step, this implies that w can be derived according to G iff it can be derived according to G'.

Consider now the case of k-limited systems. Let $L \in \mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{E[P]T0L})$. By the AFL properties[3] of $\mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{E[P]T0L})$, L can be represented as $L = \bigcup_{a \in \Delta}\{w \in \Delta^* \mid wa \in L\}\{a\} \cup L_{fin}$, where L_{fin} is a finite language and each language $\{w \in \Delta^* \mid wa \in L\}$ lies in $\mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{E[P]T0L})$.

Since $\mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{E[P]0L})$ is easily seen to be closed under union, it suffices to show how a valence test k-limited E[P]T0L system $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$ with the set of tables $\mathfrak{P} = \{P_1, \ldots, P_m\}$ can be transformed into a system G_a generating $L(G)\{a\}$.

The k-limited EP0L system G_a is defined as follows: $G_a = (\Sigma \cup \{\$\}, \Delta, P, \omega\$)$, where

$$
\begin{aligned}
P &= \bigcup_{i=1}^{m}\{(a \to w, rp_i^{-1}) \mid (a \to w, r) \in P_i\} \\
&\quad \cup \bigcup_{i=1}^{m}\{(\$ \to \$, p_i^j) \mid 0 \le j \le |\Sigma|k - 1\} \\
&\quad \cup \bigcup_{i=1}^{m}\{(\$ \to a, p_i^j) \mid 0 \le j \le |\Sigma|k - 1\}.
\end{aligned}
$$

Again, it is easy to see by induction that a word αa is generated in G_a iff $\alpha\$$ is generated in G_a iff α is generated in G. $\square$

7 Closure properties

Firstly, we give a proof of the fact that each of the families of $uk\ell$ET0L languages is a semi-AFL. This was posed as an open question in [32]. To this end, we need the notion of a transducer.

A *one-input finite state transducer with accepting state*, or *1-a-transducer* for short, is a 6-tuple $M = (Q, X, Y, \delta, Q_0, Q_f)$, where Q is a finite set of states, X and Y are finite (input and output) alphabets, $Q_0 \subseteq Q$ is the set of initial states, and $Q_f \subseteq Q$ is the set of accepting or final states, and δ is a finite subset of $Q \times X^* \times Y^* \times Q$. M is called *λ-free* if $\delta \subseteq Q \times X^* \times Y^+ \times Q$.

By a *computation* of such a 1-a-transducer, a word $h = h_1 \cdots h_n \in \delta^+$ is understood such that

- $\mathrm{pr}_1(h_1) \in Q_0$, $\mathrm{pr}_4(h_n) \in Q_f$;
- $\forall 1 \le i \le n-1 (\mathrm{pr}_1(h_{i+1}) = \mathrm{pr}_4(h_i))$;

[3] For notions concerning classification of language families, cf. Section 7.

where pr_i are homomorphisms on δ^* defined by

$$\mathrm{pr}_i((x_1, x_2, x_3, x_4)) = x_i \quad \text{for} \quad i = 1, 2, 3, 4.$$

The set of all computations of M is denoted by $C(M)$. A *1-a-transducer mapping* or *rational transduction* is defined for each language $L \subseteq X^*$ by $M(L) = \mathrm{pr}_3(\mathrm{pr}_2^{-1}(L) \cap C(M))$.

A *generalized sequential machine (gsm)* is a 1-a-transducer with transition relation $\delta \subset (Q \times X \times Y^* \times Q)$.

Remark 7.1 By Nivat's theorem [1, 15], a family of languages $\mathcal{L}$ is a trio if and only if it is closed under λ-free 1-a-transducer mappings. $\mathcal{L}$ is a full trio iff it is closed under 1-a-transducer mappings.

It is easily seen that every rational transduction can be defined by a finite transducer in *normal form*, i.e., with transition relation $\delta \subset Q \times (X \cup \{\lambda\}) \times (Y \cup \{\lambda\}) \times Q$.

Let us further mention without proof[4] that every λ-free rational transduction is representable as the composition $T = T_2 T_1$ of a λ-free "normal form" transducer $\mathcal{A}_1$ followed by a restricted erasing T_2, where a *k-restricted erasing* is a finite-state 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

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

Recall the following notions:

- A (full) semi-AFL is a (full) trio which is closed under union.
- A full AFL (where AFL abbreviates "**A**bstract **F**amily of **L**anguages") is a full semi-AFL which is closed under Kleene star and catenation.
- An AFL is a semi-AFL which is closed under Kleene plus and catenation.

Theorem 7.2 Let $k \in \mathbb{N}$. Then, we can prove the following facts:

1. $\mathcal{L}(uk\ell\mathrm{ET0L})$ is a catenation-closed full semi-AFL.
2. $\mathcal{L}(uk\ell\mathrm{EPT0L})$ is closed under union, catenation, and non-erasing gsm mappings, i.e., especially under intersection with regular sets and under non-erasing homomorphism.

[4] We refer to [9].

3. $\mathcal{L}(uk\ell\text{E[P]0L})$ is closed under union, and (non-erasing) gsm mappings, i.e., especially under intersection with regular sets and under (non-erasing) homomorphism.

Proof. First, we consider systems with possibly more than one table. Let us mention that there is a variant of uniformly limited systems, called exact uniformly limited systems in [8], where we require that exactly k occurrences of symbols are rewritten in each derivation step; if the string is shorter than k, the derivation stops (here, we need a finite set of initial words). Both variants are equivalent as regards their generating power, see [8, pages 57ff.]. We use the exact variant in the following.

First, we show that $\mathcal{L}(uk\ell\text{ET0L})$ is a full trio. Hence, we have to show the closure of $\mathcal{L}(uk\ell\text{ET0L})$ under 1-a-transducer mappings. So, let $G = (\Sigma, \Delta, H, \Omega, k)$ be an exact $uk\ell$ET0L system and $M = (Q, \Delta, \Delta', \delta, Q_0, Q_f)$ be a 1-a-transducer in normal form.

We construct an exact $uk\ell$ET0L system $G' = (\Sigma', \Delta', H', \Omega', k)$ such that $L(G') = M(L(G))$. Firstly, we set $\Omega = \Omega_1 \cup \Omega_2$, where

$$\begin{aligned}
\Omega_1 &= \{(q_0, w_1, q_1) \cdots (q_{m-1}, w_m, q_m) \mid \\
&\qquad \forall 1 \le i \le m (w_i \in \Sigma, w_1 \cdots w_m \in \Omega, q_i \in Q), q_0 \in Q_0, q_m \in Q_f\} \quad \text{and} \\
\Omega_2 &= \{w \in M(L(G)) \mid |w| < k\}.
\end{aligned}$$

The proof idea is a modification of the classical triple construction. Let

$$\begin{aligned}
\overline{\Delta} &= \{\overline{(q, a, q')} \mid q, q' \in Q, a \in \Delta \cup \{\lambda\}\} \text{ and} \\
\Sigma' &= (Q \times (\Sigma \cup \{\lambda\}) \times Q) \cup \overline{\Delta} \cup \Delta' \cup \{F\}.
\end{aligned}$$

For each table $h \in H$, we put a table h' into H', with

$$h' = \bigcup_{a \to w \in h} M'(a \to w) \cup X_F, \quad \text{where}$$

$$\begin{aligned}
M'(a \to w) &= \{(q_1, a, q_{n+1}) \to (q_1, w_1, q_2) \cdots (q_n, w_n, q_{n+1}) \mid \\
&\qquad w = w_1 \cdots w_n, |w| = n \ge 1, w_i \in \Sigma, q_i \in Q\}, \\
M'(a \to \lambda) &= \{(q, a, q) \to \lambda \mid q \in Q\}, \text{ and} \\
X_F &= \{X \to F \mid X \in \Sigma' \setminus (Q \times \Sigma \times Q)\}.
\end{aligned}$$

Furthermore, we have a table $h_\lambda \in H'$ responsible for the handling of empty inputs:

$$\begin{aligned}
h_\lambda &= \{x \to x \mid x \in \Sigma'\} \cup \\
&\cup \{\{(q, B, q'') \to (q, B, q')(q', \lambda, q''), (q, B, q'') \to (q, \lambda, q')(q', B, q'') \mid \\
&\qquad q, q', q'' \in Q, B \in \Sigma \cup \{\lambda\}\}
\end{aligned}$$

The termination table h_t consists of productions

$$\begin{aligned} h_t &= \{(q,a,q') \to \overline{(q,a,q')}, \overline{(q,a,q')} \to \overline{(q,a,q')}, \overline{(q,a,q')} \to w \\ &\quad \mid (q,a,w,q') \in \delta, a \in \Delta \cup \{\lambda\}\} \cup \\ &\cup \quad \{X \to F \mid X \in \Sigma'\}. \end{aligned}$$

A correct simulation can be done as follows:

1. A sequence of table applications in G can be emulated by the corresponding sequence of table applications in G', where table h' in G' plays the role of table h in G.

   Observe that the length of the words derived in this manner is the same when comparing the derivations in G and G', which means that the same applicability condition (imposed by the exact mode restriction) is met.

2. Empty inputs may now be intercalated if necessary by using table h_λ.

3. Finally, table h_t mimicks the transducer application. Here, the symbols from $\overline{\Delta}$ are necessary in the case when the word with which we enter the termination phase has a length not divisible by k.

Of course, it is possible to apply tables h_λ and h_t before completing the sequence of h' table applications. This causes no problem as long as the number of occurrences of symbols from $Q\times\Sigma\times Q$ is at least k. If that number of occurrences is less than k (but the length of the word is greater than k), rules from X_F will introduce a failure symbol, and the derivation is blocked in this way.

Now, it is an easy exercise to alter our construction above in order to show that $uk\ell$EPT0L languages are closed under non-erasing gsm mappings.

For the closure under union, we refer to [32, Theorem 5.1].

We turn to the closure under catenation. Let

$$G^{(i)} = (\Sigma^{(i)}, \Delta, H^{(i)}, \{\omega_1^{(i)}, \ldots, \omega_{n^{(i)}}^{(i)}\}, k)$$

(for $1 \le i \le 2$) be two exact $uk\ell$ET0L systems. Without loss of generality, we can assume that $\Sigma^{(1)} \cap \Sigma^{(2)} = \Delta$. Moreover, we may suppose that both systems are pseudo-synchronized (as defined in [32] in the case of $uk\ell$ET0L systems; their proof transfers to exact $uk\ell$ET0L systems), meaning that terminal symbols only derive a special failure symbol.

For every table $h \in H^{(i)}$, define $h' = h \cup \{X \to F, X \in \Sigma^{(3-i)} \cup \{F\}\}$, where $F \notin \Sigma^{(1)} \cup \Sigma^{(2)}$, and let $H = \{h' \mid h \in H^{(1)} \cup H^{(2)}\}$. Furthermore, define

$$\Omega = \{\omega_i\omega_j \mid 1 \le i \le n^{(1)}, 1 \le j \le n^{(2)}\}.$$

Then, $G = (\Sigma^{(1)} \cup \Sigma^{(2)} \cup \{F\}, \Delta, H, \Omega, k)$ is an exact $uk\ell$ET0L systems which generates $L(G^{(1)}) \cdot L(G^{(2)})$.

Now, we consider uniformly limited E0L systems. As shown in [32, Theorem 5.1], the corresponding language classes are closed under homomorphism and union. Moreover, it is shown in [32, Theorem 6.1] that $uk\ell$E0L systems are equivalent to so-called k-context-free languages as defined in [25], which basically correspond to pseudo-synchronized exact $uk\ell$E0L systems as introduced above. Now, it is an easy exercise to alter our construction above in order to show that $uk\ell$E[P]0L languages are closed under (non-erasing) gsm mappings. A gsm makes no λ-moves. Therefore, we need not introduce the tables h_λ and h_t to cope with this case. The obedience of the gsm transfer rules is directly checked in the rules of $M'(a \to w)$. □

Open Problem 8. Unfortunately, we were not able to show that $uk\ell$EPT0L form a trio; by the remark above, we would have to show basically closure under restricted erasing in that case. □

Open Problem 9. So, $uk\ell$ET0L systems enjoy the same natural closure properties as permitting random context grammars or programmed grammars without appearance checking [4]. In each of these cases, we conjecture that the corresponding language class is not closed under Kleene star, but we have no proof for this claim.

Let us mention that the main reason why we think that $\mathcal{L}(uk\ell\text{ET0L})$ is not closed under Kleene star (if $k > 1$) is the completeness condition imposed on the tables. On the other hand, it is quite easy to show that "incomplete" $uk\ell$ET0L systems are closed under star operation. □

By extending the transition relation δ of a 1-a-transducer to incorporate monoid elements from M, i.e., $\delta \subset Q \times X^* \times Y^* \times Q \times M$, we obtain the notion of a *valence 1-a-transducer*. A sequence $h = h_1 \cdots h_n \in \delta^+$ of such a machine is a computation if, in addition to the requirements of a transducer, we have $1 = \mathrm{pr}_5(h_1) \cdots \mathrm{pr}_5(h_n)$ (computed within M). In particular, we have defined a *valence generalized sequential machine* slightly different from [20].

The following fact is easily seen (in fact, the corresponding statement in the non-valence case readily transfers):

Lemma 7.3 If a language family $\mathcal{L}$ is closed under valence gsm mappings, then it is closed under intersection with regular valence languages. □

As regards the standard closure properties, for systems with valuations, we have the following three theorems:

Theorem 7.4 Let $k \in \mathbb{N}$.

1. $\mathcal{L}(\mathrm{Val}', \mathrm{E[P][T]0L})$ is closed under union and intersection with regular sets.
2. $\mathcal{L}(\mathrm{Val}', k\ell\mathrm{E[P][T]0L})$ is closed under union.
3. $\mathcal{L}(\mathrm{Val}', uk\ell\mathrm{E[P][T]0L})$ is closed under union and intersection with regular sets.

Proof. The following observation is helpful. If there is a constructive proof for a closure property which does not change the terminal alphabet (as it is the case for union and intersection with regular sets) for the underlying non-evaluated class, then the same proof is also valid in the valuated case, since this does not interfere with the additional test of $\varphi(w) = 1$. Therefore, the reader is referred to [24, Theorem I.7], [26, Theorem 4.14] and [32, Theorem 5.1]. □

Observe that in the case of valuated systems, we obtain a number of positive closure results for non-extended systems via Theorem 6.1, which is quite a rare situation in the case of pure L systems and its variants.

Open Problem 10. Unfortunately, we were not able to prove other closure properties under the AFL operations. □

Theorem 7.5 Let $k \in \mathbb{N}$.

1. $\mathcal{L}(\mathrm{Val}, \mathrm{ET0L})$ is a catenation-closed full semi-AFL. Moreover, it is closed under valence 1-a-transducer mappings.
2. $\mathcal{L}(\mathrm{Val}, \mathrm{E0L})$ is closed under union, catenation and gsm mappings. Moreover, it is closed under valence gsm mappings.
3. $\mathcal{L}(\mathrm{Val}_t, \mathrm{ET0L})$ is a full AFL.
4. $\mathcal{L}(\mathrm{Val}, k\ell\mathrm{E[T]0L}) = \mathcal{L}(\mathrm{Val}_t, k\ell\mathrm{E[T]0L})$ is a full AFL. Moreover, it is closed under valence 1-a-transducer mappings.
5. $\mathcal{L}(\mathrm{Val}, uk\ell\mathrm{ET0L})$ is a catenation-closed full semi-AFL. Moreover, it is closed under valence 1-a-transducer mappings.
6. $\mathcal{L}(\mathrm{Val}, uk\ell\mathrm{E0L})$ is closed under union, catenation and gsm mappings. Moreover, it is closed under valence gsm mappings.
7. $\mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{ET0L})$ is a catenation-closed full semi-AFL.
8. $\mathcal{L}(\mathrm{Val}_t, uk\ell\mathrm{E0L})$ is closed under union and gsm mappings.

Similar closure properties hold for the classes formed by propagating systems, but we do not know whether $\mathcal{L}(\mathrm{Val}, uk\ell\mathrm{EPT0L})$ is closed under inverse morphisms. Hence, we can merely show that $\mathcal{L}(\mathrm{Val}, uk\ell\mathrm{EP[T]0L})$ is closed under union, catenation and non-erasing valence gsm mappings.

Proof. The claimed (semi-)AFL properties are simply inherited from the known (semi-)AFL properties of programmed and context-free languages, see [4], and our previous Theorems 3.2, 3.5, 3.6, and 6.3.

In the case of valence systems, consider the following arguments:

- The case of union can be treated by using a standard construction.
- Similarly, a standard proof (which uses disjoint alphabets) showing the closure of the underlying non-valence class under catenation readily transfers to the valence case; we use disjoint alphabets and disjoint prime number sets and decorate the rules with the corresponding valences.
- Analogously, a standard proof showing the closure of the underlying non-valence class under gsm mappings or rational transductions (as the one in the proof of Theorem 7.2) can be easily adapted to incorporate valences.

Moreover, a standard proof showing the closure under gsm mappings can be easily adapted to show the closure under valence gsm mappings. For example, in Theorem 7.2, one has to decorate the rules in table h_t with the corresponding valences.

The last assertion follows by combining Theorems 4.6 and 7.2.

Concerning propagating systems, nearly the same argument applies. The difficulty concerning uniformly limited systems can be seen in the discussion following Theorem 7.2. □

Note that the closure of ET0L languages under Kleene star does not guarantee the closure of the corresponding class of valence ET0L languages under Kleene star, since it is impossible to make an arbitrarily large number of (intermediate) one-tests. Since the necessary amount of such intermediate one-tests cannot be predetermined in general, the trick using disjoint prime-number sets in order to defer tests to the end of the whole derivation (see our argument on catenation) does not work here.

Open Problem 11. We do not have non-closure results regarding AFL-operations for any of the families listed in the previous theorem, so that there are still lots of open questions in this context. □

Our previous lemma yields:

Corollary 7.6 Let $k \in \mathbb{N}$. Every class $\mathcal{L}(\mathrm{Val}, [(u)k\ell]\mathrm{E}[\mathrm{P}][\mathrm{T}]0\mathrm{L})$ is closed under intersection with regular valence languages. □

Theorem 7.7 Let $k \in \mathbb{N}$. $\mathcal{L}(\mathrm{Val}, [(u)k\ell]\mathrm{E}[\mathrm{T}]0\mathrm{L})$ is the closure of $\mathcal{L}([(u)k\ell]\mathrm{E}[\mathrm{T}]0\mathrm{L})$ under valence gsm mappings.

Similar closure properties do not seem to hold for the classes formed by propagating systems.

Proof. From the previous theorem, we know that each of the language classes $\mathcal{L}([(u)k\ell]\mathrm{E}[\mathrm{T}]0\mathrm{L})$ is closed under valence gsm mappings. On the other hand, it is possible that every applied rule makes its value explicit by deriving a special additional symbol. The valence gsm uses these symbols in order to check the valence property, yielding the empty word when reading such a special symbol.[5] □

Next, we are going to show that the introduction of valuations/valences does increase the generative power of L systems. We do this by showing the following interesting closure property of valence E[T]0L systems.

Theorem 7.8 The families $\mathcal{L}(\mathrm{Val}, \mathrm{E}[\mathrm{T}]0\mathrm{L})$ are closed under permutations.

Proof. Note that $\mathrm{Perm}(L)$ is the image of $L \subseteq \Sigma^*$ under a certain valence gsm mapping. More specifically, if $\Sigma = \{a_1, \ldots, a_k\}$, the valence gsm contains one state z and all valence transitions $(z, a_i, a_j, z, p_i \cdot p_j^{-1})$ where $p_1, \ldots, p_k$ are pairwise different primes. □

The non-closure of E[T]0L languages under permutations was not explicitly shown in [24], but it follows immediately from some other results mentioned in that book.

Theorem 7.9 The families $\mathcal{L}(\mathrm{E}[\mathrm{T}]0\mathrm{L})$ and $\mathcal{L}(\mathrm{Val}', \mathrm{E}[\mathrm{T}]0\mathrm{L})$ are not closed under permutations.

Proof. Note that the language $L_1 = \{a^{2^n} b^m \mid n, m \geq 0\}$ is an E0L language. However, $\mathrm{Perm}(L_1) = \{w \in \{a,b\}^* \mid |w|_a = 2^n, n \geq 0\}$ is not in $\mathcal{L}(\mathrm{E0L})$, see [24].

For the case of ET0L languages, consider the ET0L language $L_2 = \{a^{2^n} b^{3^n} \mid n \in \mathbb{N}\}$. $\mathrm{Perm}(L_2) = \{w \in \{a,b\}^* \mid |w|_a = 2^n, |w|_b = 3^n \geq 0\}$ is not in $\mathcal{L}(\mathrm{ET0L})$, see [24, page 249].

Finally, it can be shown that the permutations of the languages L_1 and L_2 cannot be generated using a valuated E[T]0L system. Assume, for contradiction,

[5] In the case of regular or context-free (valence) grammars, the corresponding construction is exhibited in [20, Theorem 1].

that $\mathrm{Perm}(L_1) = L(G)$ for some valuated E0L system $G = (\Sigma, \{a, b\}, \mathfrak{P}, \omega, \varphi)$. From $a \in L_1$, we get that $\varphi(a) = 1$. Together with $ab \in L_1$, this yields $\varphi(b) = 1$. Hence, $L(G) = L(G')$ where G' is the underlying E0L system of G. The proof of $\mathrm{Perm}(L_2) \notin \mathcal{L}(\mathrm{Val}', \mathrm{ET0L})$ is done analogously. □

Let us remark that sometimes in the literature, the operation quasiintersection is considered, see [4, page 53], where

$$Qi(L_1, L_2) = L_1 \cap \mathrm{Perm}(L_2)\,.$$

Since $Qi(\Delta^*, L_2) = \mathrm{Perm}(L_2)$ for any language $L_2 \subseteq \Delta^*$, we have:

Corollary 7.10 The families $\mathcal{L}(\mathrm{E[T]0L})$ and $\mathcal{L}(\mathrm{Val}', \mathrm{ET[0]L})$ are not closed under quasiintersection. □

Theorem 7.11 Let $k \in \mathbb{N}$. $\mathcal{L}(\mathrm{Val}, [(u)k\ell]\mathrm{E[D][T]0L})$ is the homomorphic closure of $\mathcal{L}(\mathrm{Val}', [(u)k\ell]\mathrm{E[D][T]0L})$.

Proof. The inclusions themselves are shown in Theorems 4.4 and 7.5 .

Let $G = (\Sigma, \Delta, \mathfrak{P}, \omega)$ be a valence ET0L system. Let Q be the set of all positive rational numbers appearing as a label in some rule of $\mathfrak{P}$. Now, we define the valuated ET0L system $G' = (\Sigma', \Delta', \mathfrak{P}', \omega, \varphi)$, where $\Sigma' = \Sigma \cup Q$, $\Delta' = \Delta \cup Q$, $\mathfrak{P}'$ contains, for any $P \in \mathfrak{P}$, the table

$$P' = \{a \to wq \mid (a \to w, q) \in P, a \in \Sigma, w \in \Sigma^*, q \in Q\},$$

and the valuation mapping is defined as $\varphi(a) = 1$, for $a \in \Delta$, $\varphi(q) = q$, for $q \in Q$.

Obviously, $h(L(G')) = L(G)$ where $h : (\Delta')^* \to \Delta^*$ is the homomorphism defined by $h(a) = a$, for $a \in \Delta$, $h(q) = \lambda$, for $q \in Q$.

The same proof works for (uniformly) limited systems as well. □

With the last two results, we obtain (since $\mathcal{L}(\mathrm{ET0L})$ is closed under morphisms):

Corollary 7.12 $\mathcal{L}(\mathrm{Val}', \mathrm{E[T]0L})$ is not closed under homomorphism. □

Therefore, we have:

Corollary 7.13

$$\mathcal{L}(\mathrm{E[T]0L}) \subset \mathcal{L}(\mathrm{Val}', \mathrm{E[T]0L}) \subset \mathcal{L}(\mathrm{Val}, \mathrm{E[T]0L}) \subseteq \mathcal{L}(\mathrm{Val}_t, \mathrm{ET0L})\,. \quad \square$$

Open Problem 12. Of course, the question arises whether the last inclusion is strict or not. Specifically, it would be interesting to know whether the word problem is solvable for $\mathcal{L}(\text{Val}, \text{E[T]0L})$. It is solvable for $\mathcal{L}(\text{E[T]0L})$ and, therefore, it is also solvable for $\mathcal{L}(\text{Val}', \text{E[T]0L})$. □

Together with Theorem 7.7, we get another non-closure property of ET0L languages which has not been observed before in the literature.

Corollary 7.14 $\mathcal{L}((\text{Val}'), \text{E[T]0L})$ is not closed under valence gsm mappings.□

We have already shown that the closure under valence gsm mappings implies the closure under intersection with regular valence languages. Under certain conditions, the converse is true, too.

Theorem 7.15 Let $\mathcal{L}$ be a language class closed under arbitrary gsm mappings. Then, $\mathcal{L}$ is closed under valence gsm mappings if and only if $\mathcal{L}$ is closed under intersection with regular valence languages.

Proof. Assume that $\mathcal{L}$ is closed under intersection with regular valence languages. A valence gsm g (when applied to a language $L \in \mathcal{L}$) can be simulated as follows: (a) It is possible to construct a gsm g_1 which works like g (without valences) but additionally outputs in every step a coding of the valence within a special alphabet. (b) Consider the intersection of $g_1(L)$ with the trivial regular valence language R defined such that the coded valences of the given valence gsm evaluate to 1. (c) Now, take the trivial gsm g_2 which simply discards the valence codes. Then, $g_2(g_1(L) \cap R) = g(L)$. □

Corollary 7.16 $\mathcal{L}(\text{E[T]0L})$ is not closed under intersection with regular valence languages. □

8 Conclusions and open problems

In this paper, we started investigations on parallel systems with different variants of valence regulations. Lots of things remain to be done and clarified. We indicated a number of open problems throughout the paper. Here, we put emphasis on four areas:

- We do not know whether table valences add to the power of (uniformly) limited E[P]T0L systems (with the trivial exception that $u1\ell$E[P]T0L systems with table valences (characterizing context-free valence languages) are more powerful than $u1\ell$E[P]T0L systems (characterizing context-free languages)).

- Also in the case of valence and valuated (uniformly) limited ET0L systems, many questions remain open.
- Control monoids different from $\mathbb{Q}_+$ and $\mathbb{Z}$ remain unexplored, e.g., what about $\mathbb{Z}^n$ instead of $\mathbb{Q}_+$ as valuations?
- A systematic study of "classical" restrictions of [(uniformly) limited] ET0L systems (like determinism, propagating, limitations on the number of tables) in the presence of the different notions of valence regulations has to be undertaken.

References

[1] J. Berstel. *Transductions and Context-Free Languages*, volume 38 of *LAMM.* Stuttgart: Teubner, 1979.

[2] J. Dassow. Grammars with valuations — a discrete model for self-organization of biopolymers. *Discrete Applied Mathematics*, 4:161–174, 1982.

[3] J. Dassow. A remark on limited 0L systems. *J. Inf. Process. Cybern. EIK (formerly Elektron. Inf.verarb. Kybern.)*, 24(6):287–291, 1988.

[4] J. Dassow and Gh. Păun. *Regulated Rewriting in Formal Language Theory*, volume 18 of *EATCS Monographs in Theoretical Computer Science.* Berlin: Springer, 1989.

[5] L. E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with r distinct prime factors. *Amer. Journal Math.*, 35:413–422, 1903.

[6] H. Fernau. Membership for 1-limited ET0L languages is not decidable. *J. Inf. Process. Cybern. EIK (formerly Elektron. Inf.verarb. Kybern.)*, 30(4):191–211, 1994.

[7] H. Fernau. On unconditional transfer. In W. Penczek and A. Szałas, editors, *MFCS'96*, volume 1113 of *LNCS*, pages 348–359, 1996.

[8] H. Fernau and H. Bordihn. Remarks on accepting parallel systems. *International Journal of Computer Mathematics*, 56:51–67, 1995.

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

[10] H. Fernau and R. Stiebe. Valence grammars and their relatives In preparation for publication, 2000.

[11] H. Fernau and R. Stiebe. Remarks on regulated valence grammars. Unpublished Manuscript, started in 1997.

[12] 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, Berlin: Springer, 1997.

[13] H. Fernau and R. Stiebe. Regulation by valences. Technical Report No. 36(1998), Martin-Luther-Universität Halle-Wittenberg, Fachbereich Mathematik und Informatik, 1998. (extended version of [12])

[14] H. Fernau and D. Wätjen. Remarks on regulated limited ET0L systems and regulated context-free grammars. *Theoretical Computer Science*, 194:35–55, 1998.

[15] S. Ginsburg. *Algebraic and Automata-Theoretic Properties of Formal Languages*. North-Holland, Amsterdam, 1975.

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

[17] D. Hauschildt and M. Jantzen. Petri net algorithms in the theory of matrix grammars. *Acta Informatica*, 31:719–728, 1994.

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

[19] N. D. Jones and S. Skyum. Complexity of some problems concerning L systems. *Mathematical Systems Theory*, 13:29–43, 1979.

[20] M. Marcus and Gh. Păun. Valence gsm-mappings. *Bull. Math. Soc. Sci. Math. Roumanie*, 31(3):219–229, 1987.

[21] A. Meduna. Evaluated grammars. *Acta Cybernetica*, 8:169–176, 1987.

[22] Gh. Păun. A new generative device: valence grammars. *Rev. Roumaine Math. Pures Appl.*, XXV(6):911–924, 1980.

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

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

[25] K. Salomaa. Hierarchy of k-context-free languages. *International Journal of Computer Mathematics*, 26:69–90,193–205, 1989.

[26] D. Wätjen. k-limited 0L systems and languages. *J. Inf. Process. Cybern. EIK (formerly Elektron. Inf.verarb. Kybern.)*, 24(6):267–285, 1988.

[27] D. Wätjen. Regulation of k-limited ET0L systems. *International Journal of Computer Mathematics*, 47:29–41, 1993.

[28] D. Wätjen. Regulation of uniformly k-limited T0L systems. *J. Inf. Process. Cybern. EIK (formerly Elektron. Inf.verarb. Kybern.)*, 30(3):169–187, 1994.

[29] D. Wätjen. On regularly controlled k-limited T0L systems. *International Journal of Computer Mathematics*, 55:57–66, 1995.

[30] D. Wätjen. Regulations of uniformly k-limited ET0L systems and their relations to controlled context-free grammars. *Journal of Automata, Languages and Combinatorics*, 1(1):55–74, 1996.

[31] D. Wätjen and E. Unruh. On the degree of synchronization of klET0L systems. *Information Processing Letters*, 29:87–89, 1988.

[32] D. Wätjen and E. Unruh. On extended k-uniformly-limited T0L systems and languages. *J. Inf. Process. Cybern. EIK (formerly Elektron. Inf.verarb. Kybern.)*, 26(5/6):283–299, 1990.