On the Balancedness of Tree-to-Word Transducers

Raphaela Löbel(✉), Michael Luttenberger, and Helmut Seidl

TU München, Munich, Germany
{loebel,luttenbe,seidl}@in.tum.de

Abstract. A language over an alphabet $\mathsf{B} = \mathsf{A} \cup \overline{\mathsf{A}}$ of opening (A) and closing ($\overline{\mathsf{A}}$) brackets, is balanced if it is a subset of the Dyck language $\mathbb{D}_\mathsf{B}$ over B, and it is well-formed if all words are prefixes of words in $\mathbb{D}_\mathsf{B}$. We show that well-formedness of a context-free language is decidable in polynomial time, and that the longest common reduced suffix can be computed in polynomial time. With this at a hand we decide for the class 2-TW of non-linear tree transducers with output alphabet B^* whether or not the output language is balanced.

Keywords: Balancedness of tree-to-word transducer · Equivalence · Longest common suffix/prefix of a CFG

1 Introduction

Structured text requires that pairs of opening and closing brackets are properly nested. This applies to text representing program code as well as to XML or HTML documents. Subsequently, we call properly nested words over an alphabet B of opening and closing brackets *balanced*. Balanced words, i.e. structured text, need not necessarily be constructed in a structured way. Therefore, it is a non-trivial problem whether the set of words produced by some kind of text processor, consists of balanced words only. For the case of a single pair of brackets and context-free languages, decidability of this problem has been settled by Knuth [3] where a polynomial time algorithm is presented by Minamide and Tozawa [9]. Recently, these results were generalized to the output languages of monadic second-order logic (MSO) definable tree-to-word transductions [8]. The case when the alphabet B consists of *multiple* pairs of brackets, though, seems to be more intricate. Still, balancedness for context-free languages was shown to be decidable by Berstel and Boasson [1] where a polynomial time algorithm again has been provided by Tozawa and Minamide [13]. Whether or not these results for B can be generalized to MSO definable transductions as e.g. done by finite copying macro tree transducers with regular look-ahead, remains as an open problem. Reynier and Talbot [10] considered visibly pushdown transducers and showed decidability of this class with well-nested output in polynomial time.

Here, we provide a first step to answering this question. We consider deterministic tree-to-word transducers which process their input at most twice by calling in their axioms at most two *linear* transductions of the input. Let 2-TW denote the class of these transductions. Note that the output languages of *linear* deterministic tree-to-word transducers is context-free, which does not need to be the case for 2-TW transducers. 2-TW forms a subclass of MSO definable transductions which allows to specify transductions such as *prepending* an XML document with the list of its section headings, or *appending* such a document with the list of figure titles. For 2-TW transducers we show that balancedness is decidable—and this in polynomial time. In order to obtain this result, we first generalize the notion of balancedness to the notion of *well-formedness* of a language, which means that each word is a *prefix* of a balanced word. Then we show that well-formedness for context-free languages is decidable in polynomial time. A central ingredient is the computation of the *longest common suffix* of a context-free language L over B *after reduction* i.e. after canceling all pairs of matching brackets. While the proof shares many ideas with the computation of the longest common prefix of a context-free language [7] we could not directly make use of the results of [7] s.t. the results of this paper fully subsume the results of [7]. Now assume that we have verified that the output language of the first linear transduction called in the axiom of the 2-TW transducer and the *inverted* output language of the second linear transformation both are well-formed. Then balancedness of the 2-TW transducer in question, effectively reduces to the *equivalence* of two deterministic linear tree-to-word transducers—modulo the reduction of opening followed by corresponding closing brackets. Due to the well-formedness we can use the equivalence of linear tree-to-word transducers over the *free group* which can be decided in polynomial time [5].

This paper is organized as follows. After introducing basic concepts in Sect. 2, Sect. 3 shows how balancedness for 2-TW transducers can be reduced to equivalence over the free group and well-formedness of LT_Δs. Section 4 considers the problem of deciding well-formedness of context-free languages in general.

Missing proofs can be found in the extended version of this paper [4].

2 Preliminaries

As usual, $\mathbb{N}$ ($\mathbb{N}_0$) denotes the natural numbers (including 0). The power set of a set S is denoted by 2^S. Σ denotes some generic (nonempty) alphabet, Σ^* and Σ^ω denote the set of all finite words and the set of all infinite words, respectively. Then $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ is the set of all countable words. Note, that the transducers considered here output finite words only; however, for the operations needed to analyze the output infinite words are very helpful. We denote the empty word by ε. For a finite word $w = w_0 \dots w_l$, its reverse w^R is defined by $w^R = w_l \dots w_1 w_0$; as usual, set $L^R := \{w^R \mid w \in L\}$ for $L \subseteq \Sigma^*$. A is used to denote an alphabet of *opening brackets* with $\overline{\mathsf{A}} = \{\overline{a} \mid a \in \mathsf{A}\}$ the derived alphabet of *closing brackets*, and $\mathsf{B} := \mathsf{A} \cup \overline{\mathsf{A}}$ the resulting alphabet of *opening and closing brackets*.

Longest Common Prefix and Suffix. Let Σ be an alphabet. We first define the *longest common prefix* of a language, and then reduce the definition of the *longest* *common suffix* to it by means of the reverse. We write $\overset{p}{\sqsubseteq}$ to denote the prefix relation on $\varSigma^{\infty}$, i.e. we have $u \overset{p}{\sqsubseteq} w$ if either (i) $u, w \in \varSigma^*$ and there exists $v \in \varSigma^*$ s.t. $w = uv$, or (ii) $u \in \varSigma^*$ and $w \in \varSigma^{\omega}$ and there exists $v \in \varSigma^{\omega}$ s.t. $w = uv$, or (iii) $u, w \in \varSigma^{\omega}$ and $u = w$. We extend $\varSigma^{\infty}$ by a greatest element $\top \notin \varSigma^{\infty}$ w.r.t. $\overset{p}{\sqsubseteq}$ s.t. $u \overset{p}{\sqsubseteq} \top$ for all $u \in \varSigma^{\infty}_{\top} := \varSigma^{\infty} \cup \{\top\}$. Then every set $L \subseteq \varSigma^{\infty}_{\top}$ has an infimum w.r.t. $\overset{p}{\sqsubseteq}$ which is called the *longest common prefix* of L, abbreviated by $\mathsf{lcp}(L)$. Further, define $\varepsilon^{\omega} := \top$, $\top^R := \top$, and $\top w := \top =: w\top$ for all $w \in \varSigma^{\infty}_{\top}$.

In Sect. 4 we will need to study the *longest common suffix (*lcs*)* of a language L. For $L \subseteq \varSigma^*$, we can simply set $\mathsf{lcs}(L) := \mathsf{lcp}(L^R)^R$, but also certain infinite words are very useful for describing how the lcs changes when concatenating two languages (see e.g. Example 2). Recall that for $u, w \in \varSigma^*$ and $w \neq \varepsilon$ the ω-regular expression uw^{ω} denotes the unique infinite word $uwww\ldots$ in $\bigcap_{k \in \mathbb{N}_0} uw^k \varSigma^{\omega}$; such a word is also called *ultimately periodic*. For the lcs we will use the expression $w^{\mho}u$ to denote the *ultimately left-periodic* word $\ldots wwwu$ that ends on the suffix u with infinitely many copies of w left of u; these words are used to abbreviate the fact that we can generate a word $w^k u$ for unbounded $k \in \mathbb{N}_0$. As we reduce the lcs to the lcp by means of the reverse, we define the reverse of $w^{\mho}u$, denoted by $(w^{\mho}u)^R$, by means of $(w^{\mho}u)^R := u^R(w^R)^{\omega}$.

Definition 1. *Let $\varSigma^{\mathsf{ulp}}$ denote the set of all expressions of the form $w^{\mho}u$ with $u \in \varSigma^*$ and $w \in \varSigma^+$. $\varSigma^{\mathsf{ulp}}$ is called the set of* ultimately left-periodic words. *Define the reverse of an expression $w^{\mho}u \in \varSigma^{\mathsf{ulp}}$ by means of $(w^{\mho}u)^R := u^R(w^R)^{\omega}$. Accordingly, set $(uw^{\omega})^R := (w^R)^{\mho}u^R$ for $u \in \varSigma^*$, $w \in \varSigma^+$.*

The suffix order *on $\varSigma^* \cup \varSigma^{\mathsf{ulp}} \cup \{\top\}$ is defined by $u \overset{s}{\sqsubseteq} v :\Leftrightarrow u^R \overset{p}{\sqsubseteq} v^R$. The* longest common suffix (lcs) *of a language $L \subseteq \varSigma^* \cup \varSigma^{\mathsf{ulp}}$ is $\mathsf{lcs}(L) := \mathsf{lcp}(L^R)^R$.*

For instance, we have $\mathsf{lcs}((bba)^{\mho}, (ba)^{\mho}a) = a$, and $\mathsf{lcs}((ab)^{\mho}, (ba)^{\mho}b) = (ab)^{\mho}$.

As usual, we write $u \overset{s}{\sqsubset} v$ if $u \overset{s}{\sqsubseteq} v$, but $u \neq v$. As the lcp is the infimum w.r.t. $\overset{p}{\sqsubseteq}$, we also have for $x, y, z \in \{\top\} \cup \varSigma^* \cup \varSigma^{\mathsf{ulp}}$ and $L, L' \subseteq \{\top\} \cup \varSigma^* \cup \varSigma^{\mathsf{ulp}}$ that (i) $\mathsf{lcs}(x, y) = \mathsf{lcs}(y, x)$, (ii) $\mathsf{lcs}(x, \mathsf{lcs}(y, z)) = \mathsf{lcs}(x, y, z)$, (iii) $\mathsf{lcs}(L) \overset{s}{\sqsubseteq} \mathsf{lcs}(L')$ for $L \supseteq L'$, and (iv) $\mathsf{lcs}(Lx) = \mathsf{lcs}(L)x$ for $x \in \{\top\} \cup \varSigma^*$. In Lemma 8 in the appendix of the extended version [4] we derive further equalities for lcs that allow to simplify its computation. In particular, the following two equalities (for $x, y \in \varSigma^*$) are very useful:

$$\mathsf{lcs}(x, xy) = \mathsf{lcs}(x, y^{\mho}) = \mathsf{lcs}(x, xy^k) \quad \text{for every } k \geq 1$$

$$\mathsf{lcs}(x^{\mho}, y^{\mho}) = \begin{cases} (xy)^{\mho} & \text{if } xy = yx \\ \mathsf{lcs}(xy, x^{\mho}) = \mathsf{lcs}(xy, yx^k) & \text{if } xy \neq yx, \text{ for every } k \geq 1 \end{cases}$$

For instance, we have $\mathsf{lcs}((ab)^{\mho}, (bab)^{\mho}) = bab = \mathsf{lcs}(abbab, (ab)^{\mho})$. Note also that by definition we have $\varepsilon^{\mho} = \top$ s.t. $\mathsf{lcs}(x^{\mho}, \varepsilon^{\mho}) = (x\varepsilon)^{\mho}$. We will use the following observation frequently:

Lemma 1. *Let $L \subseteq \Sigma^*$ be nonempty. Then for any $x \in L$ we have $\mathit{lcs}(L) = \mathit{lcs}(\mathit{lcs}(x,z) \mid z \in L)$; in particular, there is some* witness *$y \in L$ (w.r.t. x) s.t. $\mathit{lcs}(L) = \mathit{lcs}(x,y)$.*

Involutive Monoid. We briefly recall the basic definitions and properties of the finitely generated involutive monoid, but refer the reader for details and a formal treatment to e.g. [11]. Let A be a finite alphabet (of opening brackets/letters). From A we derive the alphabet $\overline{\mathsf{A}} := \{\overline{a} \mid a \in \mathsf{A}\}$ (of closing brackets/letters) where we assume that $\mathsf{A} \cap \overline{\mathsf{A}} = \emptyset$. Set $\mathsf{B} := \mathsf{A} \cup \overline{\mathsf{A}}$. We use roman letters $p, q, \ldots$ to denote words over A, while Greek letters $\alpha, \beta, \gamma, \ldots$ will denote words over B.

We extend $\bar{\cdot}$ to an involution on B^* by means of $\overline{\varepsilon} := \varepsilon$, $\overline{\overline{a}} := a$ for all $a \in \mathsf{A}$, and $\overline{\alpha\beta} := \overline{\beta}\,\overline{\alpha}$ for all other $\alpha, \beta \in \mathsf{B}^*$. Let $\xrightarrow{\rho}$ be the binary relation on B^* defined by $\alpha a \overline{a} \beta \xrightarrow{\rho} \alpha\beta$ for any $\alpha, \beta \in \mathsf{B}^*$ and $a \in \mathsf{A}$, i.e. $\xrightarrow{\rho}$ cancels nondeterministically one pair of matching opening and closing brackets. A word $\alpha \in \mathsf{B}^*$ is *reduced* if it does not contain any infix of the form $a\overline{a}$ for any $a \in \mathsf{A}$, i.e. α is reduced if and only if it has no direct successor w.r.t. $\xrightarrow{\rho}$. For every $\alpha \in \mathsf{B}^*$ canceling all matching brackets in any arbitrary order always results in the same unique reduced word which we denote by $\rho(\alpha)$; we write $\alpha \overset{\rho}{=} \beta$ if $\rho(\alpha) = \rho(\beta)$. Then $\mathsf{B}^*/\overset{\rho}{=}$ is the free involutive monoid generated by A, and $\rho(\alpha)$ is the shortest word in the $\overset{\rho}{=}$-equivalence class of α. For $L \subseteq \mathsf{B}^*$ we set $\rho(L) := \{\rho(w) \mid w \in L\}$.

Well-Formed Languages and Context-Free Grammars. We are specifically interested in context-free grammars (CFG) G over the alphabet B. We write $\rightarrow_G$ for the rewrite rules of G. We assume that G is reduced to the productive nonterminals that are reachable from its axiom S. For simplicity, we assume for the proofs and constructions that the rules of G are of the form

$$X \rightarrow_G YZ \qquad X \rightarrow_G Y \qquad X \rightarrow_G \overline{u}\,v$$

for nonterminals X, Y, Z and $u, v \in \mathsf{A}^*$. We write $L_X := \{\alpha \in \mathsf{B}^* \mid X \rightarrow_G^* \alpha\}$ for the language generated by the nonterminal X. Specifically for the axiom S of G we set $L := L_S$. The height of a derivation tree w.r.t. G is measured in the maximal number of nonterminals occurring along a path from the root to any leaf, i.e. in our case any derivation tree has height at least 1. We write $L_X^{\leq h}$ for the subset of L_X of words that possess a derivation tree of height at most h s.t.:

$$L_X^{\leq 1} = \{\overline{u}\,v \mid X \rightarrow_G \overline{u}\,v\} \quad L_X^{\leq h+2} = L_X^{\leq h+1} \cup \bigcup_{X \rightarrow_G YZ} L_Y^{\leq h+1} L_Z^{\leq h+1} \cup \bigcup_{X \rightarrow_G Y} L_Y^{\leq h+1}$$

We will also write $L_X^{<h}$ for $L_X^{\leq h-1}$ and $L_X^{=h}$ for $L_X^{\leq h} \setminus L_X^{<h}$. The *prefix closure* of $L \subseteq \mathsf{B}^*$ is denoted by $\mathsf{Prf}(L) := \{\alpha' \mid \alpha'\alpha'' \in L\}$.

Definition 2. *Let $\alpha \in B^*$ and $L \subseteq B^*$.*

1. *Let $\Delta(\alpha) := |\alpha|_A - |\alpha|_{\overline{A}}$ be the difference of opening brackets to closing brackets. α is* nonnegative *if $\forall \alpha' \overset{p}{\sqsubseteq} \alpha \colon \Delta(\alpha') \geq 0$. $L \subseteq B^*$ is* nonnegative *if every $\alpha \in L$ is nonnegative.*

2. *A context-free grammar G with $L(G) \subseteq \mathsf{B}^*$ is* nonnegative *if $L(G)$ is nonnegative. For a nonterminal X of G let $d_X := \sup(\{-\Delta(\alpha') \mid \alpha'\alpha'' \in L_X\})$.*
3. *A word α is* well-formed (short: wwf) resp. well-formed (short: wf) *if $\rho(\alpha) \in \overline{\mathsf{A}}^*\mathsf{A}^*$ resp. if $\rho(\alpha) \in \mathsf{A}^*$. A context-free grammar G is wf if $L(G)$ is wf. $L \subseteq \mathsf{B}^*$ is wwf* resp. *wf if every word of L is wwf resp. wf.*
4. *A context-free grammar G is* bounded well-formed (bwf) *if it is wwf and for every nonterminal X there is a (shortest) word $r_X \in \mathsf{A}^*$ with $|r_X| = d_X$ s.t. $r_X L_X$ is wf.*

Note that $d_X \geq 0$ as we can always choose $\alpha' = \varepsilon$ in the definition of d_X.

As already mentioned in the abstract and the introduction, we have that L is wf *iff* $\mathsf{Prf}(L)$ is wf *iff* L is a subset of the prefix closure of the Dyck language generated by $S \to \varepsilon$, $S \to SS$, $S \to aS\overline{a}$ (for $a \in \mathsf{A}$). We state some further direct consequences of above definition: (i) L is nonnegative *iff* the image of L under the homomorphism that collapses A to a singleton is wf. Hence, if L is wf, then L is nonnegative. Δ is an ω-continuous homomorphism from the language semiring generated by B to the tropical semiring $\langle \mathbb{Z} \cup \{-\infty\}, \min, +\rangle$. Thus it is decidable in polynomial time if G is nonnegative using the Bellman-Ford algorithm [2]. (ii) If L is not wf, then there exists some $\alpha \in \mathsf{Prf}(L) \setminus \{\varepsilon\}$ s.t. $\Delta(\alpha) < 0$ or $\alpha \overset{\rho}{=} ua\overline{b}$ for $u \in \mathsf{A}^*$ and $a, b \in \mathsf{A}$ (with $a \neq b$). (iii) If L_X is wwf, then $d_X = \sup\{|y| \mid \gamma \in L_X, \rho(\gamma) = \overline{y}\, z\}$.

In particular, because of context-freeness, it follows that, if G is wf, then for every nonterminal X there is $r_X \in \mathsf{A}^*$ s.t. (i) $\overline{r_X} \in \rho(\mathsf{Prf}(L_X))$, (ii) $|r_X| = d_X$ and (iii) $r_X L_X$ is wf. Hence:

Lemma 2. *A context-free grammar G is wf iff G is bwf with $r_S = \varepsilon$ for S the axiom of G.*

The words r_X mentioned in the definition of bounded well-formedness can be computed in polynomial time using the Bellman-Ford algorithm similar to [13]; more precisely, a *straight-line program (*SLP*)* (see e.g. [6] for more details on SLPs), i.e. a context-free grammar generating exactly one derivation tree and thus word, can be extracted from G for each r_X.

Lemma 3. *Let $L = L(G)$ be wf. Let X be some nonterminal of G. Let $r_X \in \mathsf{A}^*$ be the shortest word s.t. $r_X L_X$ is wf. We can compute an* SLP *for r_X from G in polynomial time.*

Tree-to-Word Transducers. We define a *linear* tree-to-word transducer ($\mathsf{LT_B}$) $M = (\Sigma, \mathsf{B}, Q, S, R)$ where Σ is a finite ranked input alphabet, B is the finite (unranked) output alphabet, Q is a finite set of states, the axiom S is of the form u_0 or $u_0 q(x_1) u_1$ with $u_0, u_1 \in \mathsf{B}^*$ and R is a set of rules of the form $q(f(x_1, \ldots, x_m)) \to u_0 q_1(x_{\sigma(1)}) u_1 \ldots q_n(s_{\sigma(n)}) u_n$ with $q, q_i \in Q$, $f \in \Sigma$, $u_i \in \mathsf{B}^*$, $n \leq m$ and σ an injective mapping from $\{1, \ldots, n\}$ to $\{1, \ldots, m\}$. Since non-deterministic choices of linear transducers can be encoded into the input symbols, we may, w.l.o.g., consider *deterministic* transducers only. For simplicity, we moreover assume the transducers to be *total.* This restriction can be lifted by additionally taking a top-down deterministic tree automaton for the domain into account. The constructions introduced in Sect. 3 would then have to be applied w.r.t. such a domain tree automaton. As we consider total deterministic transducers there is exactly one rule for each pair $q \in Q$ and $f \in \Sigma$.

A 2-copy tree-to-word transducer (2-TW) is a tuple $N = (\Sigma, \mathsf{B}, Q, S, R)$ that is defined in the same way as an $\mathsf{LT_B}$ but the axiom S is of the form u_0 or $u_0 q_1(x_1) u_1 q_2(x_1) u_2$, with $u_i \in \mathsf{B}^*$.

$\mathcal{T}_\Sigma$ denotes the set of all trees/terms over Σ. We define the semantics $[\![q]\!] : \mathcal{T}_\Sigma \to \mathsf{B}^*$ of a state q with rule $q(f(t_1, \ldots, t_m)) \to u_0 q_1(t_{\sigma(1)}) u_1 \ldots q_n(t_{\sigma(n)}) u_n$ inductively by

$$[\![q]\!](f(t_1, \ldots, t_m)) = \rho(u_0 [\![q_1]\!](t_{\sigma(1)}) u_1 \ldots [\![q_n]\!](t_{\sigma(n)}) u_n)$$

The semantics $[\![M]\!]$ of an $\mathsf{LT_B}$ M with axiom u_0 is given by $\rho(u_0)$; if the axiom is of the form $u_0 q(x_1) u_1$ it is defined by $\rho(u_0 [\![q]\!](t) u_1)$ for all $t \in \mathcal{T}_\Sigma$; while the semantics $[\![N]\!]$ of a 2-TW N with axiom u_0 is again given by $\rho(u_0)$ and for axiom $u_0 q_1(x_1) u_1 q_2(x_1) u_2$ it is defined by $\rho(u_0 [\![q_1]\!](t) u_1 [\![q_2]\!](t) u_2)$ for all $t \in \mathcal{T}_\Sigma$. For a state q we define the output language $\mathcal{L}(q) = \{[\![q]\!](t) \mid t \in \mathcal{T}_\Sigma\}$; For a 2-TW M we let $\mathcal{L}(M) = \{[\![M]\!](t) \mid t \in \mathcal{T}_\Sigma\}$. Note that the output language of an $\mathsf{LT_B}$ is context-free and a corresponding context-free grammar for this language can directly read from the rules of the transducer.

Additionally, we may assume w.l.o.g. that all states q of an $\mathsf{LT_B}$ are *non-singleton*, i.e., $\mathcal{L}(q)$ contains at least two words. We call a 2-TW M *balanced* if $\mathcal{L}(M) = \{\varepsilon\}$. We say an $\mathsf{LT_B}$ M is *well-formed* if $\mathcal{L}(M) \subseteq \mathsf{A}^*$. Balanced and well-formed states are defined analogously. We use $\overline{q}$ to denote the inverse transduction of q which is obtained from a copy of the transitions reachable from q by involution of the right-hand side of each rule. As a consequence, $[\![\overline{q}\,]\!](t) = \overline{[\![q]\!](t)}$ for all $t \in \mathcal{T}_\Sigma$, and thus, $\mathcal{L}(\overline{q}\,) = \overline{\mathcal{L}(q)}$. We say that two states q, q' are *equivalent* iff for all $t \in \mathcal{T}_\Sigma$, $[\![q]\!](t) = [\![q']\!](t)$. Accordingly, two 2-TWs M, M' are equivalent iff for all $t \in \mathcal{T}_\Sigma$, $[\![M]\!](t) = [\![M']\!](t)$.

3 Balancedness of 2-TWs

Let M denote a 2-TW. W.l.o.g., we assume that the axiom of M is of the form $q_1(x_1) q_2(x_1)$ for two states q_1, q_2. If this is not yet the case, an equivalent 2-TW with this property can be constructed in polynomial time. We reduce balancedness of M to decision problems for *linear* tree-to-word transducers alone.

Proposition 1. *The 2-TW M is balanced iff the following two properties hold:*

- *Both $\mathcal{L}(q_1)$ and $\overline{\mathcal{L}(q_2)}$ are well-formed;*
- *q_1 and $\overline{q_2}$ are equivalent.*

Proof. Assume first that M with axiom $q_1(x_1) q_2(x_1)$ is balanced, i.e., $\mathcal{L}(M) = \varepsilon$. Then for all w', w'' with $w = w'w'' \in \mathcal{L}(M)$, $\rho(w') = u \in \mathsf{A}^*$ and $\rho(w'') = \overline{u}$. Thus, both $\mathcal{L}(q_1)$ and $\overline{\mathcal{L}(q_2)}$ consist of well-formed words only. Assume for a contradiction that q_1 and $\overline{q_2}$ are not equivalent. Then there is some $t \in \mathcal{T}_\Sigma$ such that $[\![q_1]\!](t) \not\overset{\rho}{=} [\![\overline{q_2}]\!](t)$. Let $[\![q_1]\!](t) = u \in \mathsf{A}^*$ and $[\![\overline{q_2}]\!](t) = \overline{[\![q_2]\!](t)} = v$ with $v \in \mathsf{A}^*$ and $u \neq v$. Then $\rho([\![q_1]\!](t)[\![q_2]\!](t)) = \rho(u\overline{v}) \neq \varepsilon$ as $u \neq v$, $u, v \in \mathsf{A}^*$. Since M is balanced, this is not possible.

Now, assume that $\mathcal{L}(q_1)$ and $\overline{\mathcal{L}(q_2)}$ are well-formed, i.e., for all $t \in \mathcal{T}_\Sigma$, $[\![q_1]\!](t) \in \mathsf{A}^*$ and $[\![q_2]\!](t) \in \overline{\mathsf{A}}^*$. Additionally assume that q_1 and $\overline{q_2}$ are equivalent, i.e., for all $t \in \mathcal{T}_\Sigma$, $[\![q_1]\!](t) = [\![\overline{q_2}]\!](t) = \overline{[\![q_2]\!](t)}$. Therefore for all $t \in \mathcal{T}_\Sigma$, $[\![q_2]\!](t) = \overline{[\![q_1]\!](t)}$ and hence,

$$\rho([\![q_1]\!](t)[\![q_2]\!](t)) = \rho([\![q_1]\!](t)\overline{[\![q_1]\!](t)}) = \varepsilon$$

Therefore, the 2-TW M must be balanced. □

The output languages of states q_1 and $\overline{q_2}$ are generated by means of context-free grammars of polynomial size.

Example 1. Consider LT_B M with input alphabet $\Sigma = \{f^{(2)}, g^{(0)}\}$ (the superscript denotes the rank), output alphabet $\mathsf{B} = \{a, \overline{a}\}$, axiom $q_3(x_1)$ and rules

$$\begin{array}{ll} q_3(f(x_1,x_2)) \to aq_2(x_1)q_2(x_2)\overline{a} & q_2(g) \to \varepsilon \\ q_2(f(x_1,x_2)) \to aq_1(x_1)q_1(x_2)\overline{a} & q_2(g) \to \varepsilon \\ q_1(f(x_1,x_2)) \to q_3(x_1)q_3(x_2) & q_1(g) \to aa \end{array}$$

We obtain a CFG producing exactly the output language of M by nondeterministically guessing the input symbol, i.e. the state q_i becomes the nonterminal W_i. The axiom of this CFG is then W_3, and as rules we obtain

$$W_3 \to aW_2W_2\overline{a} \mid \varepsilon \quad W_2 \to aW_1W_1\overline{a} \mid \varepsilon \quad W_1 \to W_3W_3 \mid aa$$

Note that the rules of M and the associated CFG use a form of iterated squaring, i.e. $W_3 \to^2 W_3^4$, that allows to encode potentially exponentially large outputs within the rules (see also Example 4). In general, words thus have to be stored in compressed form as SLPs [6].

Therefore, Theorem 2 of Sect. 4 implies that well-formedness of q_1, $\overline{q_2}$ can be decided in polynomial time. Accordingly, it remains to consider the equivalence problem for well-formed LT_Bs. Since the two transducers in question are well-formed, they are equivalent as LT_Bs iff they are equivalent when their outputs are considered over the free group $\mathcal{F}_\mathsf{A}$. In the free group $\mathcal{F}_\mathsf{A}$, we additionally have that $\overline{a}\,a \overset{\rho}{=} \varepsilon$—which does not hold in our rewriting system. If sets $\mathcal{L}(q_1), \mathcal{L}(\overline{q_2})$ of outputs for q_1 and $\overline{q_2}$, however, are well-formed, it follows for all $u \in \mathcal{L}(q_1), v \in \mathcal{L}(\overline{q_2})$ that $\rho(u\overline{v}) = \rho(\rho(u)\rho(\overline{v}))$ cannot contain $\overline{a}\,a$. Therefore, $\rho(u\overline{v}) = \varepsilon$ iff $u\overline{v}$ is equivalent to ε over the free group $\mathcal{F}_\mathsf{A}$. In [5, Theorem 2], we have proven that equivalence of LT_Bs where the output is interpreted over the free group, is decidable in polynomial time. Thus, we obtain our main theorem.

Theorem 1. *Balancedness of 2-*TW*s is decidable in polynomial time.*

4 Deciding Well-Formedness of Context-Free Grammars

As described in the preceding sections, given a 2-TW we split it into the two underlying $\mathsf{LT_B}$s that process a copy of the input tree. We then check that each of these two $\mathsf{LT_B}$s are equivalent w.r.t. the free group. As sketched in Example 1 we obtain a context-free grammar for the output language of each of these $\mathsf{LT_B}$s. It then remains to check that both context-free grammars are well-formed. In order to prove that we can decide in polynomial time whether a context-free grammar is well-formed (short: wf), we proceed as follows:

First, we introduce in Definition 3 the *maximal suffix extension* of a language $L \subseteq \Sigma^*$ w.r.t. the lcs (denoted by $\mathsf{lcsx}(L)$), i.e. the longest word $u \in \Sigma^\infty$ s.t. $\mathsf{lcs}(uL) = u\,\mathsf{lcs}(L)$. We then show that the relation $L \approx_{\mathsf{lcs}} L' :\Leftrightarrow \mathsf{lcs}(L) = \mathsf{lcs}(L') \wedge \mathsf{lcsx}(L) = \mathsf{lcsx}(L')$ is an equivalence relation on Σ^* that respects both union and concatenation of languages (see Lemma 5). It then follows that for every language $L \subseteq \Sigma^*$ there is some subset $\mathsf{T_{lcs}}(L) \subseteq L$ of size at most 3 with $L \approx_{\mathsf{lcs}} \mathsf{T_{lcs}}(L)$.

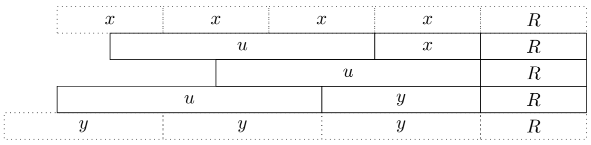
We then use $\mathsf{T_{lcs}}$ to compute a finite $\approx_{\mathsf{lcs}}$-equivalent representation $\mathsf{T}_X^{\leq h}$ of the *reduced* language generated by each nonterminal X of the given context-free grammar inductively for increasing derivation height h. In particular, we show that we only have to compute up to derivation height $4N + 1$ (with N the number of nonterminals) in order to decide whether G is wf: In Lemma 7 we show that, if G is wf, then we have to have $\mathsf{T}_X^{\leq 4N+1} \approx_{\mathsf{lcs}} \mathsf{T}_X^{\leq 4N}$ for all nonterminals X of G. The complementary result is then shown in Lemma 6, i.e. if G is not wf, then we either cannot compute up to $\mathsf{T}_X^{\leq 4N+1}$ as we discover some word that is not wf, or we have $\mathsf{T}_X^{\leq 4N} \not\approx_{\mathsf{lcs}} \mathsf{T}_X^{\leq 4N+1}$ for at least one nonterminal X.

Maximal Suffix Extension and lcs-Equivalence. We first show that we can compute the longest common suffix of the union $L \cup L'$ and the concatenation LL' of two languages $L, L' \subseteq \Sigma^*$ if we know both $\mathsf{lcs}(L)$ and $\mathsf{lcs}(L')$, and in addition, the longest word $\mathsf{lcsx}(L)$ resp. $\mathsf{lcsx}(L')$ by which we can extend $\mathsf{lcs}(L)$ resp. $\mathsf{lcs}(L')$ when concatenating another language from left. In contrast to the computation of the lcp presented in [7], we have to take the maximal extension lcsx explicitly into account. In this paragraph we do not consider the involution, thus let Σ denote an arbitrary alphabet.

Definition 3. *For $L \subseteq \Sigma^*$ with $R = \mathsf{lcs}(L)$ the* maximal suffix extension (lcsx) *of L is defined by $\mathsf{lcsx}(L) := \mathsf{lcs}(z^\frown \mid zR \in L)$.*

Recall that by definition $\mathsf{lcsx}(\emptyset) = \mathsf{lcs}(\emptyset) = \top$ and $\mathsf{lcsx}(\{R\}) = \mathsf{lcs}(\varepsilon^\frown) = \top$. The following example motivates the definition of lcsx:

Example 2. Consider the language $L = \{R, xR, yR\}$ with $\mathsf{lcs}(L) = R$ and $\mathsf{lcsx}(L) = \mathsf{lcs}(x^\frown, y^\frown)$. Assume we prepend some word $u \in \Sigma^*$ to L resulting in the language $uL = \{uR, uxR, uyR\}$, see the following picture for an illustration (dotted boxes represent copies of $z \in \{x, y\}$ stemming from the usual line of argumentation that, if z is a suffix of $u = u'z$, then $uzR = u'zzR$, and thus eventually covering all of u by $z^\frown$):



As motivated by the picture, $\mathsf{lcs}(uL)$ is given by $\mathsf{lcs}(u, x^{\frown}, y^{\frown})R$. Using the concept of ultimately left-periodic words, we may also formalize this as follows:

$$\begin{aligned}
\mathsf{lcs}(u\{xR, yR, R\}) &= \mathsf{lcs}(u, ux, uy)R \\
&= \mathsf{lcs}(\mathsf{lcs}(u, ux), \mathsf{lcs}(u, uy))R \quad (\text{as } \mathsf{lcs}(u, ux) = \mathsf{lcs}(u, x^{\frown})) \\
&= \mathsf{lcs}(\mathsf{lcs}(u, x^{\frown}), \mathsf{lcs}(u, y^{\frown}))R \\
&= \mathsf{lcs}(u, \mathsf{lcs}(x^{\frown}, y^{\frown}))R \qquad = \mathsf{lcs}(u, \mathsf{lcsx}(L))\mathsf{lcs}(L)
\end{aligned}$$

In particular, if $xy = yx$, we can extend lcs by any finite suffix of $\mathsf{lcsx}(L) = (xy)^{\frown}$ (note that, if $x = \varepsilon = y$, then $\mathsf{lcsx}(L) = \varepsilon^{\frown} = \top$ is defined to be the greatest element w.r.t. $\overset{s}{\sqsubseteq}$); but if $xy \neq yx$, we can extend it at most to $\mathsf{lcsx}(L) = \mathsf{lcs}(x^{\frown}, y^{\frown}) = \mathsf{lcs}(xy, yx) \overset{s}{\sqsubset} xy$. Essentially, only three cases can arise as illustrated by the following three examples:

First, consider $L_1 = \{ab, cb\}$ with $\mathsf{lcs}(L_1) = b$. Obviously, for every word $u \in \Sigma^*$ we have that $\mathsf{lcs}(uL_1) = \mathsf{lcs}(L_1)$ and so we should have $\mathsf{lcsx}(L_1) = \varepsilon$. Instantiating the definition we obtain indeed $\mathsf{lcsx}(L_1) = \mathsf{lcs}(a^{\frown}, b^{\frown}) = \mathsf{lcs}(\varepsilon) = \varepsilon$.

As another example consider $L_2 = \{a, baa\}$ with $\mathsf{lcs}(L_2) = a$. Here, we obtain $\mathsf{lcsx}(L_2) = \mathsf{lcs}(\varepsilon^{\frown}, (ba)^{\frown}) = \mathsf{lcs}(\top, (ba)^{\frown}) = (ba)^{\frown}$, i.e. the suffix of L_2 can be extended by any finite suffix of $(ba)^{\frown} = \ldots bababa$.

Finally, consider $L_3 = \{b, ba^n b, aba^n b\}$ with $\mathsf{lcs}(L_3) = b$ for some fixed $n \in \mathbb{N}$. As mentioned in Sect. 2, we have $\mathsf{lcs}(x^{\frown}, y^{\frown}) = \mathsf{lcs}(xy, yx)$ for $xy \neq yx$. We thus obtain in this case $\mathsf{lcs}((ba^n)^{\frown}, (aba^n)^{\frown}) = \mathsf{lcs}(ba^n\, aba^n, aba^n\, ba^n) = a^n ba^n$. The classic result by Fine and Wilf states that, if $xy \neq yx$, then $|\mathsf{lcs}(x^{\frown}, y^{\frown})| < |x| + |y| - \gcd(|x|, |y|)$. Thus $x = ba^n$ and $y = aba^n$ constitute an extremal case where the lcs is only finitely extendable.

If $\mathsf{lcs}(L)$ is not contained in L, then $\mathsf{lcs}(L)$ has to be a strict suffix of every shortest word in L, and thus immediately $\mathsf{lcsx}(L) = \varepsilon$. As in the case of the lcs, also $\mathsf{lcsx}(L)$ is already defined by two words in L:

Lemma 4. *Let $L \subseteq \Sigma^*$ with $|L| \geq 2$ and $R := \mathsf{lcs}(L)$. Fix any $xR \in L \setminus \{R\}$. Then there is some $yR \in L \setminus \{R\}$ s.t. $\mathsf{lcsx}(L) = \mathsf{lcs}(x^{\frown}, y^{\frown}) = \mathsf{lcs}(x^{\frown}, y^{\frown}, z^{\frown})$ for all $zR \in L$. If $xy = yx$, then $R \in L$.*

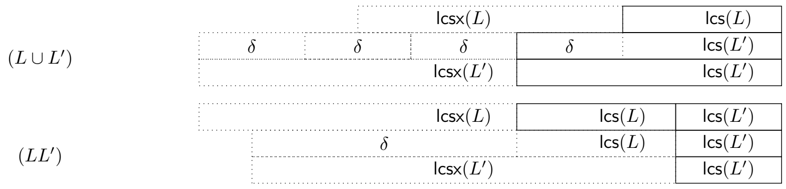
We show that we can compute the lcs and the extension lcsx of the union resp. the concatenation of two languages solely from their lcs and lcsx. To this end, we define the lcs-*summary* of a language as:

Definition 4. *For $L \subseteq \Sigma^*$ set $\pi_{\mathsf{lcs}}(L) := (\mathsf{lcs}(L), \mathsf{lcsx}(L))$. The equivalence relation $\approx_{\mathsf{lcs}}$ on 2^{Σ^*} is defined by: $L \approx_{\mathsf{lcs}} L'$ iff $\pi_{\mathsf{lcs}}(L) = \pi_{\mathsf{lcs}}(L')$.*

Lemma 5. *Let $L, L' \subseteq \Sigma^*$ with $\pi_{\mathsf{lcs}}(L) = (R, E)$ and $\pi_{\mathsf{lcs}}(L') = (R', E')$. If $L = \emptyset$ or $L' = \emptyset$, then $\pi_{\mathsf{lcs}}(L \cup L') = (\mathsf{lcs}(R, R'), \mathsf{lcs}(E, E'))$, and $\pi_{\mathsf{lcs}}(LL') = (\top, \top)$. Assume thus $L \neq \emptyset \neq L'$ which implies $R \neq \top \neq R'$. Then:*

- *$\mathsf{lcs}(L \cup L') = \mathsf{lcs}(R, R')$ and $\mathsf{lcs}(LL') = \mathsf{lcs}(R, E')R'$.*
- *If $\mathsf{lcs}(R, R') \notin \{R, R'\}$, then $\mathsf{lcsx}(L \cup L') = \varepsilon$; else w.l.o.g. $R' = \delta R$ and $\mathsf{lcsx}(L \cup L') = \mathsf{lcs}(E, \mathsf{lcs}(E', E'\delta)\delta)$.*
- *If $\mathsf{lcs}(R, E') \overset{s}{\sqsubset} R$, then $\mathsf{lcsx}(LL') = \varepsilon$; else $E' = \delta R$ and $\mathsf{lcsx}(LL') = \mathsf{lcs}(E, \delta)$.*

Example 3. Lemma 5 can be illustrated as follows:



For instance, consider $L = \{a, baa\}$ and $L' = \{aa, baaa\}$ s.t. $\pi_{\mathsf{lcs}}(L) = (a, (ba)^\omega)$ and $\pi_{\mathsf{lcs}}(L') = (aa, (ba)^\omega)$. Applying Lemma 5, we obtain for the union $\mathsf{lcs}(L \cup L') = \mathsf{lcs}\big(a, aa\big) = a$ and $\mathsf{lcsx}(L \cup L') = \mathsf{lcs}\big((ba)^\omega, \mathsf{lcs}((ba)^\omega, (ba)^\omega a)a\big) = a$. In case of the concatenation, Lemma 5 yields $\mathsf{lcs}(LL') = \mathsf{lcs}\big(a, (ba)^\omega\big)aa = aaa$ and $\mathsf{lcsx}(LL') = \mathsf{lcs}\big((ba)^\omega, (ab)^\omega\big) = \varepsilon$.

As both the lcs and the lcsx are determined by already two words (cf. Lemmas 1 and 4), it follows that every $L \subseteq \Sigma^*$ is $\approx_{\mathsf{lcs}}$-equivalent to some sublanguage $\mathsf{T}_{\mathsf{lcs}}(L) \subseteq L$ consisting of at most three words where the words xR, yR can be chosen arbitrarily up to the stated constraints (with $R = \mathsf{lcs}(L)$):

$$\mathsf{T}_{\mathsf{lcs}}(L) := \begin{cases} L & \text{if } |L| \leq 2 \\ \{R, xR, yR\} & \text{if } \{R, xR, yR\} \subseteq L \wedge \mathsf{lcsx}(L) = \mathsf{lcs}(x^\omega, y^\omega) \\ \{xR, yR\} & \text{if } R = \mathsf{lcs}(xR, yR) \wedge R \notin L \wedge \{xR, yR\} \subseteq L \end{cases}$$

Deciding Well-Formedness. For the following, we assume that G is a context-free grammar over $\mathsf{B} = \mathsf{A} \cup \overline{\mathsf{A}}$ with nonterminals $\mathfrak{X}$. Set $N := |\mathfrak{X}|$. We further assume that G is nonnegative, and that we have computed for every nonterminal X of G a word $r_X \in \mathsf{A}^*$ (represented as an SLP) s.t. $|r_X| = d_X$ and $\overline{r_X} \in \mathsf{Prf}(\rho(L_X))$.[1] In order to decide whether G is wf we compute the languages $\rho(r_X L_X^{\leq h})$ modulo $\approx_{\mathsf{lcs}}$ for increasing derivation height h using fixed-point iteration. Assume inductively that (i) $r_X L_X^{\leq h}$ is wf and (ii) that we have computed $\mathsf{T}_X^{\leq h} := \mathsf{T}_{\mathsf{lcs}}(\rho(r_X L_X^{\leq h})) \approx_{\mathsf{lcs}} \rho(r_X L_X^{\leq h})$ for all $X \in \mathfrak{X}$ up to height h. Then we can compute $\mathsf{T}_{\mathsf{lcs}}(\rho(r_X L_X^{\leq h+1}))$ for each nonterminal as follows:

$$\begin{aligned} & \rho(r_X L_X^{\leq h+1}) \\ = \;& \rho(r_X L_X^{\leq h}) \cup \textstyle\bigcup_{X \to_G Y} \rho(r_X \overline{r_Y} \;\; r_Y L_Y^{\leq h}) \cup \bigcup_{X \to_G YZ} \rho(r_X \overline{r_Y} \;\; r_Y L_Y^{\leq h} \;\; \overline{r_Z} \;\; r_Z L_Z^{\leq h}) \\ \approx_{\mathsf{lcs}} \;& \mathsf{T}_X^{\leq h} \cup \textstyle\bigcup_{X \to_G Y} \rho(r_X \overline{r_Y} \;\; \mathsf{T}_Y^{\leq h}) \cup \bigcup_{X \to_G YZ} \rho(r_X \overline{r_Y} \;\; \mathsf{T}_Y^{\leq h} \;\; \overline{r_Z} \;\; \mathsf{T}_Z^{\leq h}) \\ \approx_{\mathsf{lcs}} \;& \mathsf{T}_{\mathsf{lcs}}\Big(\rho\Big(\mathsf{T}_X^{\leq h} \cup \textstyle\bigcup_{X \to_G Y} r_X \overline{r_Y} \;\; \mathsf{T}_Y^{\leq h} \cup \bigcup_{X \to_G YZ} r_X \overline{r_Y} \;\; \mathsf{T}_Y^{\leq h} \;\; \overline{r_Z} \;\; \mathsf{T}_Z^{\leq h}\Big)\Big) =: \mathsf{T}_X^{\leq h+1} \end{aligned}$$

[1] $\overline{r_X}$ is (after reduction) a longest word of closing brackets in $\rho(L_X)$ (if G is wf, then r_X is unique). An SLP encoding r_X can be computed in polynomial time while checking that G is nonnegative; see Definition 2 and the subsequent explanations, and the proof of Lemma 3 in the appendix of the extended version [4]. All required operations on words run in time polynomial in the size of the SLPs representing the words, see e.g. [6].

Note that, if all constants $r_X\overline{r_Y}$ and all $\mathsf{T}_X^{\leq h}$ are wf, but G is not wf, then the computation has to fail while computing $r_X\overline{r_Y}\,\mathsf{T}_Y^{\leq h}\overline{r_Z}$; see the following example.

Example 4. Consider the nonnegative context-free grammar G given by the rules (with the parameter $n \in \mathbb{N}$ fixed)

$$\begin{array}{lllll} S \to Uc & U \to AV \mid W_n & V \to U\overline{B} & W_i \to W_{i-1}W_{i-1} & (2 \leq i \leq n) \\ A \to a & B \to b & \overline{B} \to \overline{b} & W_1 \to BB & \end{array}$$

with axiom S. Except for $\overline{B}$ all nonterminals generate nonnegative languages. Note that the nonterminals W_n to W_1 form an SLP that encodes the word b^{2^n} by means of iterated squaring which only becomes productive at height $h = n + 1$. For $h \geq n + 3$ we have:

$$\begin{array}{lll} L_S^{\leq h} = \{a^k b^{2^n} \overline{b}^{\,k} c \mid k \leq \lfloor \frac{h-(n+3)}{2} \rfloor\} & & \\ L_U^{\leq h} = \{a^k b^{2^n} \overline{b}^{\,k} \mid k \leq \lfloor \frac{h-(n+2)}{2} \rfloor\} & L_{W_i}^{\leq h} = \{b^{2^i}\} & L_B^{\leq h} = \{b\} \\ L_V^{\leq h} = \{a^k b^{2^n} \overline{b}^{\,k+1} \mid k \leq \lfloor \frac{h-(n+3)}{2} \rfloor\} & L_A^{\leq h} = \{a\} & L_{\overline{B}}^{\leq h} = \{\overline{b}\} \end{array}$$

Here the words r_X used to cancel the longest prefix of closing brackets (after reduction) are $r_S = r_U = r_V = r_W = r_A = r_B = \varepsilon$ and $r_{\overline{B}} = b$. Note that $r_X L_X^{\leq h}$ is wf for all nonterminals X up to $h \leq h_0 = 2^{n+1} + (n+2)$ s.t. $\mathsf{T}_{\mathsf{lcs}}(\rho(r_S L_S^{\leq h})) \approx_{\mathsf{lcs}} \mathsf{T}_S^{\leq h} = \{b^{2^n} c, a^k b^{2^n - k(h)} c\}$ for $k(h) = \lfloor (h - (n + 3))/2 \rfloor$ and $n + 3 \leq h \leq h_0$; in particular, the lcs of $\mathsf{T}_S^{\leq h}$ has already converged to c at $h = n+3$, only its maximal extension lcsx changes for $n + 3 \leq h \leq h_0$. We discover the first counterexample $a^{2^n}\overline{b}$ that G is not wf while computing $\mathsf{T}_V^{\leq h_0+1} = \mathsf{T}_{\mathsf{lcs}}(\rho(\mathsf{T}_U^{\leq h_0}\overline{b}))$.

As illustrated in Example 4, if G is not wf, then the minimal derivation height $h_0 + 1$ at which we discover a counterexample might be exponential in the size of the grammar. The following lemma states that up to this derivation height h_0 the representations $\mathsf{T}_X^{\leq h}$ cannot have converged (modulo $\approx_{\mathsf{lcs}}$).

Lemma 6. *If $L = L(G)$ is not wf, then there is some least h_0 s.t. $r_X L_Y^{\leq h_0}\overline{r_Z}$ is not wf with $X \to_G YZ$. For $h \leq h_0$, all $r_X L_X^{\leq h}$ are wf s.t. $\mathsf{T}_X^{\leq h} \approx_{\mathsf{lcs}} \rho(r_X L_X^{\leq h})$. If $h_0 \geq 4N+1$, then at least for one nonterminal X we have $\mathsf{T}_X^{\leq 4N+1} \not\approx_{\mathsf{lcs}} \mathsf{T}_X^{\leq 4N}$.*

The following Lemma 7 states the complementary result, i.e. if G is wf then the representations $\mathsf{T}_X^{\leq h}$ have converged at the latest for $h = 4N$ modulo $\approx_{\mathsf{lcs}}$. The basic idea underlying the proof of Lemma 7 is similar to [7]: we show that from every derivation tree of height at least $4N + 1$ we can construct a derivation tree of height at most $4N$ such that both trees carry the same information w.r.t. the lcs (after reduction). In contrast to [7] we need not only to show that $\mathsf{T}_X^{\leq 4N}$ has the same lcs as $\rho(r_X L_X^{\leq 4N})$, but that $\mathsf{T}_X^{\leq 4N}$ has converged modulo $\approx_{\mathsf{lcs}}$ if G is wf; to this end, we need to explicitly consider lcsx, and re-prove stronger versions of the results regarding the combinatorics on words which take the involution into account (see A.6 in the appendix of the extended version [4]).

Lemma 7. *Let G be a context-free grammar with N nonterminals and $L(G)$ be* wf*. For every nonterminal X let $r_X \in \mathsf{A}^*$ s.t. $|r_X| = d_X$ and $r_X L_X$* wf*. Then $\rho(r_X L_X) \approx_{\mathsf{lcs}} \rho(r_X L_X^{\leq 4N})$, and $\mathsf{T}_X^{\leq 4N} \approx_{\mathsf{lcs}} \mathsf{T}_X^{\leq 4N+1}$ for every nonterminal X.*

The following example sketches the main idea underlying the proof of Lemma 7.

Example 5. The central combinatorial observation[2] is that for any well-formed language $\mathcal{L} \subseteq \mathsf{B}^*$ of the form

$$\mathcal{L} = (\alpha,\beta)[(\mu_1,\nu_1) + (\mu_2,\nu_2)]^*\gamma := \{\alpha\mu_{i_1}\ldots\mu_{i_l}\gamma\nu_{i_l}\ldots\nu_{i_1}\beta \mid i_1\ldots i_l \in \{1,2\}^*\}$$

we have that its *longest common suffix after reduction* $\mathsf{lcs}^\rho(\mathcal{L}) := \mathsf{lcs}(\rho(\mathcal{L}))$ is determined by the reduced longest common suffix of $\alpha\gamma\beta$ and either $(\alpha,\beta)(\mu_i,\nu_i)\gamma = \alpha\mu_i\gamma\nu_i\beta$ or $(\alpha,\beta)(\mu_i,\nu_i)(\mu_j,\nu_j)\gamma = \alpha\mu_i\mu_j\gamma\nu_j\nu_i\beta$ for some $i \in \{1,2\}$ but *arbitrary* $j \in \{1,2\}$ in the latter case.[3]

Assume now we are given a context-free grammar G with N variables. Further assume that $L := L(G)$ is well-formed. W.l.o.g. G is in Chomsky normal form and reduced to the productive nonterminals reachable from the axiom of G. Let $L^{\leq 4N}$ denote the sublanguage of words generated by G with a derivation tree of height at most $4N$. Pick a shortest (before reduction) word $\kappa_0 \in L := L(G)$. Then there is some $\kappa_1 \in L$ with $R := \mathsf{lcs}^\rho(L) = \mathsf{lcs}^\rho(\kappa_0,\kappa_1)$; we will call any such word a *witness (w.r.t. κ_0)* in the following. If $R \in L$, then $\kappa_0 \overset{\rho}{=} R$, and any word in L is a witness. In particular, there is a witness in $L^{\leq 4N}$. So assume $R \notin L$. Then we may factorize (in a unique way) $\kappa_0 = \kappa_0'' a \kappa_0'$ and $\kappa_1 = \kappa_1'' b \kappa_1'$ such that $\rho(\kappa_0') = R = \rho(\kappa_1')$ where $a, b \in \mathsf{A}$ with $a \neq b$. Then $\rho(\kappa_0) = z_0' a R$ and $\rho(\kappa_1) = z_1' b R$. Further assume that $\kappa_1 \notin L^{\leq 4N}$, otherwise we are done. Fix any derivation tree t of κ_1, and fix within t the *main* path from the root of t to the last letter b of the suffix $b\kappa_1'$ of $\rho(\kappa_1)$ (the dotted path in Fig. 1). We may assume that any path starting at a node on this main path and then immediately turning left towards a letter within the prefix κ_1'' consists of at most N nonterminals: if any nonterminal occurs twice the induced pumping tree can be pruned without changing the suffix $b\kappa_1'$; as the resulting tree is still a valid derivation tree w.r.t. G, we obtain another witness w.r.t. κ_0. Thus consider any path (including the main path) in t that leads from its root to a letter within the suffix $b\kappa_1'$. If every such path consists of at most $3N$ nonterminals, then every path in t consists of at most $4N$ nonterminals so that $\kappa_1 \in L^{\leq 4N}$ follows. Hence, assume there is at least one such path consisting of $3N + 1$ nonterminals. Then there is some nonterminal X occurring at least four times on this path. Fix four occurrences of X and factorize κ_1 accordingly

$$\kappa_1 = u s_1 s_2 s_3 w \tau_3 \tau_2 \tau_1 v =: (u,v)(s_1,\tau_1)(s_2,\tau_2)(s_3,\tau_3)w$$

[2] This observation strengthens the combinatorial results in [7] and also allows to greatly simplify the original proof of convergence given there.

[3] To clarify notation, we set $(\alpha,\beta)(\mu,\nu) := (\alpha\mu,\nu\beta)$ and $(\alpha,\beta)\gamma := \alpha\gamma\beta$, i.e. the pair (α,β) is treated as a word with a "hole" into which the pair or word on the right-hand side is substituted.

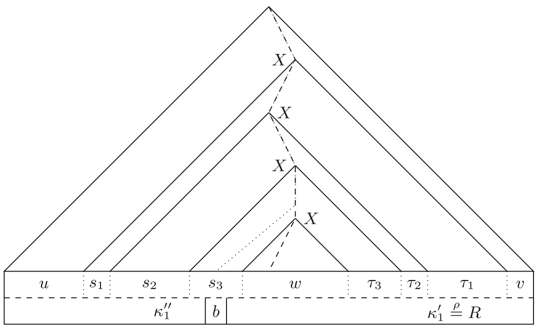


Fig. 1. Factorization of a witness $\kappa_1 = (u, v)(s_1, t_1)(s_2, t_2)(s_3, t_3)w = \kappa_1'' b\kappa_1'$ w.r.t. a nonterminal X occurring at least four times a long the dashed path in a derivation tree of κ_1 leading to a letter within the suffix $b\kappa_1'$. The dotted path depicts the main path leading to the lcs^ρ-delimiting occurrence of the letter b.

In the proof of Lemma 7 we show that we may assume—as L is well-formed—that $u, v, w, s_1, s_2, s_3 \in \mathsf{A}^*$ with only $\tau_1, \tau_2, \tau_3 \in \mathsf{B}^*$. From this factorization we obtain the sublanguage $L' := (u, v)[(s_1, \tau_1) + (s_2, \tau_2) + (s_3, \tau_3)]^*w$. Our goal is to show that $(u, v)w$ or $(u, v)(s_i, \tau_i)w$ or $(u, v)(s_i, \tau_i)(s_j, \tau_j)w$ (for $i \neq j$) is a witness w.r.t. κ_0: note that each of these words result from pruning at least one pumping tree from t which inductively leads to a procedure to reduce t to a derivation tree of height at most $4N$ that still yields a witness for $R = \mathsf{lcs}^\rho(L)$ w.r.t. κ_0. Assume thus specifically that neither $(u, v)w = uwv$ nor $(u, v)(s_3, \tau_3)w = us_3w\tau_3v$ nor $(u, v)(s_i, \tau_i)(s_3, \tau_3)w = us_is_3w\tau_3\tau_iv$ for $i \in \{1, 2\}$ is a witness w.r.t. κ_0, i.e. each of these words end on aR after reduction. Apply now the result mentioned at the beginning of this example to the language $L'' := (u, v)[(s_1, \tau_1) + (s_2, \tau_2)]^*(s_3, \tau_3)w$: by our assumptions $\kappa_1 \in L''$ is a witness w.r.t. $us_3w\tau_3v \in L''$ so that both $\mathsf{lcs}^\rho(L'') = \mathsf{lcs}^\rho(L)$ and also $(u, v)(s_1, \tau_1)^2(s_3, \tau_3)w$ is a witness w.r.t. $us_3w\tau_3v$ as we may choose $j = 1$. Thus also $\mathsf{lcs}^\rho(L) = \mathsf{lcs}^\rho(L''')$ for $L''' := (u, v)[(s_1, \tau_1) + (s_3, \tau_3)]^*w$ as $L''' \subseteq L$ and both $(u, v)w \in L'''$ and $(u, v)(s_1, \tau_1)(s_1, \tau_1)(s_3, \tau_3)w \in L'''$. Applying the same argument now to L''', but *choosing* $j \neq i$ it follows that $(u, v)(s_1, \tau_1)w$ or $(u, v)(s_3, \tau_3)(s_1, \tau_1)w$ has to be a witness w.r.t. κ_0.

The sketched argument can be adapted so that it also allows to conclude the maximal extension after reduction $\mathsf{lcsx}(\rho(L))$ has to have converged at derivation height $4N$ the latest, if L is well-formed. For details, see the proof of Lemma 7 in the appendix of the extended version [4].

As $|\mathsf{T}_X^{\leq h}| \leq 3$, a straight-forward induction also shows that every word in $\mathsf{T}_X^{\leq h}$ can be represented by an SLP that we can compute in time polynomial in G for $h \leq 4N + 1$; together with the preceding Lemmas 7 and 6 we thus obtain the main result of this section:

Theorem 2. *Given a context-free grammar G over B we can decide in time polynomial in the size of G whether G is* wf*.*

5 Conclusion

We have shown that well-formedness for context-free languages is decidable in polynomial time. This allowed us to decide in polynomial time whether or not a 2-TW is balanced. The presented techniques, however, are particularly tailored for 2-TWs. It is unclear how a generalization to transducers processing three or more copies of the input would look like. Thus, the question remains whether balancedness is decidable for general MSO definable transductions. It is also open whether even the single bracket case can be generalized beyond MSO definable transduction, e.g. to output languages of top-down tree-to-word transducers [12].

Acknowledgements. We also like to thank the anonymous reviewers for their detailed comments and valuable advice.

References

1. Berstel, J., Boasson, L.: Formal properties of XML grammars and languages. Acta Inf. **38**(9), 649–671 (2002). https://doi.org/10.1007/s00236-002-0085-4
2. Esparza, J., Kiefer, S., Luttenberger, M.: Derivation tree analysis for accelerated fixed-point computation. Theor. Comput. Sci. **412**(28), 3226–3241 (2011). https://doi.org/10.1016/j.tcs.2011.03.020
3. Knuth, D.E.: A characterization of parenthesis languages. Inf. Control **11**(3), 269–289 (1967)
4. Löbel, R., Luttenberger, M., Seidl, H.: On the balancedness of tree-to-word transducers. CoRR abs/1911.13054 (2019). http://arxiv.org/abs/1911.13054
5. Löbel, R., Luttenberger, M., Seidl, H.: Equivalence of linear tree transducers with output in the free group. CoRR abs/2001.03480 (2020). http://arxiv.org/abs/2001.03480
6. Lohrey, M.: Algorithmics on SLP-compressed strings: a survey. Groups Compl. Cryptol. **4**(2) (2012). https://doi.org/10.1515/gcc-2012-0016
7. Luttenberger, M., Palenta, R., Seidl, H.: Computing the longest common prefix of a context-free language in polynomial time. In: Niedermeier, R., Vallée, B. (eds.) STACS 2018. LIPIcs, vol. 96, pp. 48:1–48:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018). https://doi.org/10.4230/LIPIcs.STACS.2018.48
8. Maneth, S., Seidl, H.: Balancedness of MSO transductions in polynomial time. Inf. Process. Lett. **133**, 26–32 (2018). https://doi.org/10.1016/j.ipl.2018.01.002
9. Minamide, Y., Tozawa, A.: XML validation for context-free grammars. In: Kobayashi, N. (ed.) APLAS 2006. LNCS, vol. 4279, pp. 357–373. Springer, Heidelberg (2006). https://doi.org/10.1007/11924661_22
10. Reynier, P., Talbot, J.: Visibly pushdown transducers with well-nested outputs. Int. J. Found. Comput. Sci. **27**(2), 235–258 (2016). https://doi.org/10.1142/S0129054116400086
11. Sakarovitch, J.: Elements of Automata Theory. Cambridge University Press, Cambridge (2009)

12. Seidl, H., Maneth, S., Kemper, G.: Equivalence of deterministic top-down tree-to-string transducers is decidable. J. ACM **65**(4), 21:1–21:30 (2018). https://doi.org/10.1145/3182653
13. Tozawa, A., Minamide, Y.: Complexity results on balanced context-free languages. In: Seidl, H. (ed.) FoSSaCS 2007. LNCS, vol. 4423, pp. 346–360. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71389-0_25