On the Balancedness of Tree-to-word Transducers

Raphaela Löbel
TU München, Germany
raphaela.loebel@tum.de

Michael Lüttenberger
TU München, Germany
luttenbe@in.tum.de

Helmut Seidl
TU München, Germany
seidl@in.tum.de

Abstract

A language over an alphabet \( B = A \cup \overline{A} \) of opening (A) and closing (\( \overline{A} \)) brackets, is balanced if it is a subset of the Dyck language \( D_B \) over \( B \), and it is well-formed if all words are prefixes of words in \( D_B \). 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. We also show that equivalence of linear tree transducers with well-formed output in \( B^* \) is decidable in polynomial time. These two results enable us to decide in polynomial time for the class 2-TW of non-linear tree transducers with output alphabet \( B^* \) whether or not the output language is balanced.

2012 ACM Subject Classification Theory of computation → Transducers; Theory of computation → Grammars and context-free languages; Mathematics of computing → Combinatorics on words

Keywords and phrases balancedness of tree-to-word transducer, equivalence over the free group, longest common suffix/prefix of a CFG

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

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 \cite{knuth1965context} where a polynomial time algorithm is presented by Minamide and Tozawa \cite{minamide2015finite}. Recently, these results were generalized to the output languages of MSO definable tree-to-word transductions \cite{tozawa2017well}. The case when the alphabet $B$ consists of multiple pairs of brackets, though, seems to be more intricate. Still, balancedness for context-free languages could be shown to be decidable by Berstel and Boasson \cite{berstel2014context} where a polynomial time algorithm again has been provided by Tozawa and Minamide \cite{tozawa2015context}. Whether or not these results for $B$ can be generalized to MSO definable transductions remains as an open problem.

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 \cite{chomsky1963equivalence} we could not directly make use of the results of \cite{chomsky1963equivalence} s.t. the results of this paper fully subsume the results of \cite{chomsky1963equivalence}. 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. In order to decide the latter problem, we also generalize the constructions from \cite{chomsky1963equivalence} to take reduction of the output into account.

Accordingly, this paper is organized as follows. After introducing basic concepts in Section 2 Section 3 explains how to decide balancedness for $2$-$TW$ transducers, given a polynomial algorithm for well-formedness of context-free languages. In particular, it provides a reduction to the equivalence problem of well-formed linear deterministic tree transducers with output in $B^{*}$ where reductions are taken into account and provides a normal form for these. Section 4 then considers the problem of deciding well-formedness of a context-free language. It provides a summary of any such language which can be computed in polynomial time. In order to arrive at this result, rather deep insights are required into pumping properties for syntax trees with occurrences of letters and inverse letters.
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 $2^S$. $\Sigma$ denotes some generic (nonempty) alphabet, $\Sigma^*$ and $\Sigma^\omega$ 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. We denote the empty word by $\varepsilon$. For a finite word $w = w_0 \ldots w_t$, its reverse $w^R$ is defined by $w^R = w_t \ldots w_1 w_0$. $A$ is used to denote an alphabet of opening brackets with $\overline{A} = \{a \mid a \in A\}$ the derived alphabet of closing brackets, and $B := A \cup \overline{A}$ the resulting alphabet of opening and closing brackets.

Longest common prefix and suffix

Let $\Sigma$ 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 $\subseteq$ to denote the prefix relation on $\Sigma^\infty$, i.e. we have $u \subseteq v$ if either (i) $u, v \in \Sigma^*$ and there exists $v \in \Sigma^*$ s.t. $w = uv$, or (ii) $u \in \Sigma^*$ and $w \in \Sigma^\infty$ and there exists $v \in \Sigma^\omega$ s.t. $w = uv$, or (iii) $u, w \in \Sigma^\omega$ and $w = v$. We extend $\Sigma^\infty$ by a greatest element $\top \notin \Sigma^\omega$ w.r.t. $\subseteq$. Then every set $L \subseteq \Sigma^\infty$ has an infimum w.r.t. $\subseteq$, which is called the longest common prefix of $L$, abbreviated by $lcp(L)$. Further, define $\varepsilon^\omega := \top$, $\top^R := \top$, and $\top w := \top := w \top$ for all $w \in \Sigma^\infty$.

In Section 4 we will need to study the longest common suffix (lcs) of a language $L$. For $L \subseteq \Sigma^*$, we can simply set $\text{lcs}(L) := lcp(L^R)$, but also certain infinite words are very useful when studying the lcs. Recall that for $u, v \in \Sigma^*$ and $x \notin \omega$-regular expression $u w^x$ denotes the unique infinite word $u w w w \ldots$ in $\bigcap_{k \in \mathbb{N}_0} u w^x \Sigma^\omega$; such a word is also called ultimately periodic. For the lcs we will use the expression $w^x u$ to denote the “reverse” of $(u^R)(u^R)^{x}$, i.e. the infinite word $\ldots w w w w u$ 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 $k \in \mathbb{N}_0$.

Definition 1 (Ultimately left-periodic words, longest common suffix).

1. For $u \in \Sigma^*$ and $v \in \Sigma^\omega$, define the expression $w^x u$ by means of $w^x u := (u^R(w^R)^x)^R$, and its reverse by means of $(u w^x)^R = (w^R)^R u^R$. The set of ultimately left-periodic words is then $\Sigma^{ulp} := \{w^x u \mid w \in \Sigma^*, u \in \Sigma^\omega\}$.
2. The suffix order on $\Sigma^* \cup \Sigma^{ulp} \cup \{\top\}$ is then defined by $u \preceq^* v \iff u R \subseteq v R$.
3. The longest common suffix (lcs) of a language $L \subseteq \Sigma^* \cup \Sigma^{ulp}$ is then $\text{lcs}(L) := lcp(L^R) R$.

For instance, we have $\text{lcs}((b a)^N, (b a)^N a) = \text{lcp}((b a)^N, a (a b)^\omega)^R = a$, and $\text{lcs}((a b)^N, (b a)^N b) = \text{lcp}((b a)^N, b (a b)^\omega)^R = (a b)^N \in \Sigma^{ulp}$.

As usual, we write $u \preceq^* v$ if $u \preceq v$, but $u \neq v$. As the lcp is the infimum w.r.t. $\subseteq$, we also have for $x, y, z \in \{\top\} \cup \Sigma^* \cup \Sigma^{ulp}$ and $L, L' \subseteq \{\top\} \cup \Sigma^* \cup \Sigma^{ulp}$ that (i) $\text{lcs}(x, y) = \text{lcs}(y, x)$, (ii) $\text{lcs}(x, \text{lcs}(y, z)) = \text{lcs}(x, y, z)$, (iii) $\text{lcs}(L) \preceq^* \text{lcs}(L')$ for $L \supseteq L'$, and (iv) $\text{lcs}(L x) = \text{lcs}(L) x$ for $x \in \{\top\} \cup \Sigma^*$. In the appendix of the extended version (see Lemma 20) we derive further equalities for lcs that allow to simplify its computation. In particular, the following two equalities (for $x, y \in \Sigma^*$) are very useful:

\[
\begin{align*}
\text{lcs}(x, y) &= \text{lcs}(x, y^\omega) = \text{lcs}(x, xy^k) \quad \text{for every } k \geq 1 \\
\text{lcs}(x^\omega, y^\omega) &= \begin{cases} 
(x y)^\omega & \text{if } x y = y x \\
\text{lcs}(x y, x^k) & \text{if } x y \neq y x, \text{ for every } k \geq 1 
\end{cases}
\end{align*}
\]
Balancedness

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

\[ \text{Lemma 2.} \ Let L \subseteq \Sigma^* \text{ be nonempty. Then for any } x \in L \text{ we have } \text{lcs}(L) = \text{lcs}(\text{lcs}(x,z) \mid z \in L) \text{; in particular, there is some witness } y \in L \text{ (w.r.t. } x) \text{ s.t. } \text{lcs}(L) = \text{lcs}(x,y). \]

**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. [12]. Let \( A \) be a finite alphabet (of opening brackets/letters). From \( A \) we derive the alphabet \( \overline{A} := \{ \overline{a} \mid a \in A \} \) (of closing brackets/letters) where we assume that \( A \cap \overline{A} = \emptyset \). Set \( B := A \cup \overline{A} \). We will use roman letters \( p,q,\ldots,z \) to denote words over \( A \), while greek letters \( \alpha,\beta,\gamma,\ldots \) will denote words over \( B \).

We extend \( \tau \) to an involution on \( B^* \) in the usual way by means of \( \overline{a} := \varepsilon, \overline{\overline{a}} := a \) for all \( a \in A \), and \( \overline{\alpha \beta} := \overline{\beta} \overline{\alpha} \) for all other \( \alpha,\beta \in B^* \). Let \( \rightarrow_\rho \) be the rewriting system defined by \( a \alpha \beta \rightarrow_\rho \alpha \beta \) for any \( \alpha,\beta \in B^* \) and \( a \in A \). \( \rightarrow_\rho \) induces a well-founded, i.e. globally confluent and strongly normalizing, order. Given any \( \alpha \in B^* \) independent of the order in which we cancel matching brackets, we eventually arrive at the same minimal element \( \rho(\alpha) \) w.r.t. the induced order. By \( \overline{\rho} \) we denote the Shamir congruence i.e. the equivalence relation that we obtain from the set of equalities \( \{ \overline{a} = \varepsilon \mid a \in A \} \). Note that \( B^*/\overline{\rho} \) is the free involutive monoid generated by \( A \), and \( \rho(\alpha) \) is the shortest, i.e. the (maximally) reduced word in the \( \overline{\rho} \)-equivalence class of \( \alpha \). For \( L \subseteq B^* \) we set \( \rho(L) := \{ \rho(w) \mid w \in L \} \) as usual.

**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

\[
X \rightarrow_G YZ \quad X \rightarrow_G Y \quad X \rightarrow_G \overline{v}v
\]

for nonterminals \( X,Y,Z \) and \( u,v \in A^* \). We write \( L_X := \{ \alpha \in B^* \mid X \rightarrow^*_G \alpha \} \) 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 \( L^h_X \) for the subset of \( L_X \) of words that possess a derivation tree of height at most \( h \) s.t.:

\[
L^1_X = \{ \overline{v}v \mid X \rightarrow_G \overline{v}v \} \quad L^{h+2}_X = L^{h+1}_X \cup \bigcup_{X \rightarrow_G YZ} L^{h+1}_Y \cup \bigcup_{X \rightarrow_G Y} L^{h+1}_Y
\]

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

\[ \text{Definition 3.} \ Let \alpha \in B^* \text{ and } L \subseteq B^*. \]

1. Let \( \Delta(\alpha) : = |\alpha|_\Lambda - |\alpha|_{\overline{\Lambda}} \) be the difference of opening brackets to closing brackets. \( \alpha \) is nonnegative if \( \forall \alpha' \subseteq \alpha : \Delta(\alpha') \geq 0 \). \( L \subseteq B^* \) is nonnegative if every \( \alpha \in L \) is nonnegative.
2. A context-free grammar \( G \) with \( L(G) \subseteq B^* \) is nonnegative if \( L(G) \) is nonnegative. For a nonterminal \( X \) of \( G \) let \( d_X := \sup(\{ -\Delta(\alpha') \mid \alpha' \alpha'' \in L_X \} \cup \{ 0 \}) \).

2. Well-formed languages and context-free grammars
3. A word $\alpha$ is weakly well-formed (short: $wwf$) resp. well-formed (short: $wf$) if $\rho(\alpha) \in \Delta^* \Delta^*$ resp. if $\rho(\alpha) \in \Delta^*$. A context-free grammar $G$ is $wf$ if $L(G)$ is $wf$. $L \subseteq B^*$ 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 $r_X \in \Delta^*$ 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 $Prf(L)$ is $wf$ if $L$ is a subset of the prefix closure of the Dyck language generated by $S \rightarrow \varepsilon$, $S \rightarrow SS$, $S \rightarrow aS\overline{a}$ (for $a \in A$). 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. $\Delta$ is an $\omega$-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 [4]. (ii) If $L$ is not $wf$, then there exists some $\alpha \in Prf(L) \setminus \{\varepsilon\}$ s.t. $\Delta(\alpha) < 0$ or $\alpha \not= u\overline{a}b$ for $u \in \Delta^*$ and $a, b \in A$ (with $a \neq b$). (iii) If $L_X$ is $wwf$, then $d_X = \sup\{|y| \mid \gamma \in L_X, \rho(\gamma) = \overline{y}\}$.

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

$\blacktriangleright$ Lemma 4. 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 [15]; more precisely, a straight-line program (SLP) (see e.g. [7] 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 $r_X$.

$\blacktriangleright$ Lemma 5. Let $L = L(G)$ be $wf$. Let $X$ be some nonterminal of $G$. Let $r_X \in \Delta^*$ 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 (LTW) $M = (\Sigma, \Delta, Q, ax, \delta$) where $\Sigma$ is a finite ranked input alphabet, $\Delta$ is a finite (unranked) alphabet, $Q$ is a finite set of states, the axiom $ax$ is of the form $u_0 q_1(x_1) u_1$ with $u_0, u_1 \in \Delta^*$ and $\delta$ is a set of rules of the form $q(f(x_1, \ldots, x_m)) \rightarrow u_0 q_1(x_{\sigma(1)}) u_1 \ldots q_k(x_{\sigma(n)}) u_n$ with $f \in \Sigma$, $n \leq m$ and $\sigma$ a one-to-one mapping from $\{1, \ldots, n\}$ to $\{1, \ldots, m\}$. A LTW $M$ is sequential (sequential tree-to-word transducers, STW) if all rules are of the form $q(f(x_1, \ldots, x_m)) \rightarrow u_0 q_1(x_1) u_1 \ldots q_m(x_m) u_m$, i.e., $n = m$ and $\sigma(i) = i$ for all $i = 1, \ldots, m$. W.l.o.g. we assume deterministic transducers only. For simplicity, we moreover assume the transducers to be total.

This means that there is one rule for each pair $q \in Q$ and $f \in \Sigma$.

A 2-copy tree-to-word transducer (2-TW) is a tuple $N = (\Sigma, \Delta, Q, ax, \delta)$ that is defined in the same way as a LTW but the axiom is of the form $u_0 q_1(x_1) u_1 q_2(x_1) u_2$. A sequential 2-copy tree-to-word transducer (s2-TW) is a 2-TW where all rules are sequential, i.e., of the form $q(f(x_1, \ldots, x_m)) \rightarrow u_0 q_1(x_1) u_1 \ldots q_m(x_m) u_m$.

We define the semantics $[q] : T_{\Sigma} \rightarrow \Delta^*$ of a state $q$ with rule $q(f(t_1, \ldots, t_m)) \rightarrow u_0 q_1(t_{\sigma(1)}) u_1 \ldots q_n(t_{\sigma(n)}) u_n$ inductively by $[q](f(t_1, \ldots, t_m)) = u_0[q_1][t_{\sigma(1)}] u_1 \ldots [q_n](t_{\sigma(n)}) u_n$.

\[1\] In fact this restriction can be lifted by additionally taking a top-down deterministic tree automaton for the domain into account. The constructions introduced in Section 3 would then have to be applied w.r.t. such a domain tree automaton.
Balancedness

The semantics $[M]$ of a LTW $M$ with axiom $u_0 q_1 u_1$ is defined by $u_0 [q_1](t) u_1$ for all $t \in T \Sigma$; while the semantics $[N]$ of a 2-TW $N$ with axiom $u_0 q_1 x_1 u_1 q_2 x_1 u_2$ is defined by $u_0 [q_1](t_1) u_1 [q_2](t_1) u_2$ for all $t_1, t_2 \in T \Sigma$. For a state $q$ we define the output language $L(q) = \{[q](t) \mid t \in T \Sigma\}$; For a 2-TW $M$ we let $L(M) = \{[M](t) \mid t \in T \Sigma\}$. Note that the output language of a LTW is context-free and a corresponding context-free grammar for this language can directly read from the rules of the transducer.

From now on, we always consider transductions over the output alphabet $\Delta = \mathbb{B}$. Additionally, we may assume w.l.o.g. that all states $q$ of a LTW are nonsingleton, i.e., $\rho(L(q))$ contains at least two words. We call a 2-TW $M$ balanced if $\rho(L(M)) = \{\epsilon\}$. We say a LTW $M$ is well-formed if $\rho(L(M)) \subseteq \mathbb{A}^*$. Balanced and well-formed states are defined analogously. As we want to check balancedness for 2-TWs we assume for rules $q(f(x_1, \ldots, x_m)) \rightarrow \gamma_0 q_1(x_{\sigma(1)}) \gamma_1 \ldots q_n(x_{\sigma(n)}) \gamma_n$ that all $\gamma_i$ are already reduced, i.e., $\gamma_i = \pi \gamma_i v_i$, $u_i, v_i \in \mathbb{A}^*$. We use $\pi$ to denote the inverse transduction of $q$ which is obtained from a copy of the transitions reachable from $q$ by inversion of the right-hand side of each rule. As a consequence, $[\pi](t) = [\pi](t)$ for all $t \in T \Sigma$, and thus, $L(\pi) = L(q)$. We say that two states $q, q'$ are equivalent iff for all $t \in T \Sigma$, $\rho([q](t)) = \rho([q'](t))$. Accordingly, two 2-TWs $M, M'$ are equivalent iff for all $t \in T \Sigma$, $\rho([M](t)) = \rho([M'](t))$.

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 $\mathbf{ax} = q_1(x_1) q_2(x_1)$ 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 first reduce balancedness of $M$ to decision problems for linear tree-to-word transducers alone.

**Proposition 6.** The 2-TW $M$ is balanced iff the following two properties hold:

- Both $L(q_1)$ and $L(q_2)$ are well-formed;
- $q_1$ and $q_2$ are equivalent.

The output languages of states $q_1$ and $q_2$ are generated by means of context-free grammars of polynomial size. Therefore, Theorem 4 of Section 3 implies that well-formedness of $q_1, q_2$ can be decided in polynomial time. Accordingly, it remains to consider the equivalence problem for well-formed LTWs. Since the two transducers in question are well-formed, they are equivalent as LTWs iff they are equivalent when their outputs are considered over the free group. Note that in a free group $\pi a \not\equiv \pi a$ which (after reduction) contain $\pi a$. In [13], the equivalence of STWs without negated output symbols, has been reduced in polynomial-time to the morphism equivalence problem on context-free grammars — via nested word transducers. By Plandowski [11], the latter decision problem is decidable in polynomial time. Here, we present a direct reduction for same-ordered LTWs with negated output symbols to the morphism equivalence problem over the free group. Two productive LTWs $M$ and $M'$ are same-ordered if they process their output in the same order. Formally, we recursively define co-reachable states and same-ordered rules. Let $\mathbf{ax}_M = \gamma_0 q_1(x_1) \gamma_1$ and $\mathbf{ax}_{M'} = \gamma_0 q'_1(x_1) \gamma'_1$ be the axioms of two LTWs $M$ and $M'$, respectively. Then $q_1$ and $q'_1$ are co-reachable. Let $q, q'$ be two co-reachable states with rules $q(f(x_1, \ldots, x_m)) \rightarrow \gamma_0 q_1(x_{\sigma(1)}) \gamma_1 \ldots q_n(x_{\sigma(n)}) \gamma_n$ and $q'(f(x_1, \ldots, x_m)) \rightarrow \gamma'_0 q'_1(x_{\sigma'(1)}) \gamma'_1 \ldots q'_n(x_{\sigma'(n)}) \gamma'_n$, respectively. Then the two rules are same-ordered if $n = n'$ and $\sigma = \sigma'$. If the rules are same-ordered then $q_i$ and $q'_i$ are co-reachable for all $i = 1, \ldots, n$. 
Given that $M$ and $M'$ are same-ordered LTWs, we can represent the set of pairs of runs of $M$ and $M'$ by means of a single CFG $G$. $G$ has nonterminals $(q, q')$ for $q, q'$ states of $M, M'$, respectively. For the set of terminal symbols $T$ of $G$ we introduce a disjoint primed copy $B'$ of the output alphabet $B$ and let $T = B \cup B'$. Let $q(f(x_1, \ldots, x_m)) \rightarrow \gamma_0 q_1(x_{\sigma(1)}) \gamma_1 \ldots q_n(x_{\sigma(n)}) \gamma_n$ and $q'(f(x_1, \ldots, x_m)) \rightarrow \gamma_0' q_1'(x_{\sigma(1)}) \gamma_1' \ldots q_n'(x_{\sigma(n)}) \gamma_n'$ be rules of $M$ and $M'$, respectively, where $q, q'$ are co-reachable. Then we add the rule

$$(q, q') \rightarrow \gamma_0 \gamma_0'' (q_1, q_1') \gamma_1 \gamma_1'' \ldots (q_n, q_n') \gamma_n \gamma_n''$$

to $G$ where $\gamma_i''$ is obtained from $\gamma_i'$ by replacing the output symbols $a \in B$ with their primed copies $a' \in B'$. For the axioms $ax = \gamma_0 q(x_1) \gamma_1$ and $ax = \gamma_0' q'(x_1) \gamma_1'$ of $M, M'$, respectively, we introduce the start symbol $S$ in $G$ with the rule $S \rightarrow \gamma_0 \gamma_0'' (q, q') \gamma_1 \gamma_1''$ where again $\gamma_i''$ are the primed copies of $\gamma_i$. We define morphisms $f$ and $g$ by

$$f(\gamma) = \gamma \text{ if } \gamma \in B \quad g(\gamma) = \varepsilon \text{ if } \gamma \in B$$

$$f(\gamma') = \varepsilon \text{ if } \gamma' \in B' \quad g(\gamma') = \gamma \text{ if } \gamma' \in B'$$

Then $M$ and $M'$ are equivalent (with outputs interpreted over the free group) iff $g(w) = f(w)$ for all $w \in \mathcal{L}(G)$. Combining Plandowski’s polynomial construction of a test set for a context-free language to check morphism equivalence over finitely generated free groups [11], Theorem 6], with Lohrey’s polynomial algorithm for checking equivalence of SLPs over the free group [3], we obtain:

\begin{lemma}
The morphism equivalence problem of a context-free grammar in a free group is decidable in polynomial time.
\end{lemma}

As a consequence, the equivalence of same-ordered LTWs is decidable in polynomial time. Subsequently, we generalize this result to LTWs which are well-formed — but not necessarily same-ordered.

In [2], a canonical normal form for LTWs without negated output has been provided which allows to reduce equivalence of transducers to syntactic identity. That normal form, however, may increase the sizes of representations of the transducers exponentially. In order to obtain a polynomial decision procedure for equivalence, therefore, a partial normal form for LTWs without negated output symbols has been proposed [3]. We follow the latter approach and define an appropriate normal form which turns equivalent LTWs into same-ordered LTWs. The key observation in [3] is that the position of two recursive calls can be swapped provided that both produce periodic output over the same period, i.e.,

$$[q_1](t_1)[q_2](t_2) = \mathcal{L}(q_1) \cap \mathcal{L}(q_2) \subseteq w^*$$

for some $w \in \mathcal{L}(q_1)$.

In the case where words are produced in between the recursive calls, the periods of the output languages of $q_1, q_2$ may not be identical, but are at least conjugates as, e.g., in $q_1(x_1)axq_2(x_2)$ with $\mathcal{L}(q_1) \subseteq (ab)^*$ and $\mathcal{L}(q_2) \subseteq (ba)^*$. We call a well-formed LTW $M$ suffix-empty if

- for all states $q$ in $M$, $\mathcal{L}(q) \in \mathcal{A}^*$ and $\mathcal{L}(\mathcal{L}(q)) = \varepsilon$,
- for all rules $q(f(x_1, \ldots, x_n) \rightarrow v_0 q_1(x_{\sigma(1)}) \pi_1 v_1 \ldots q_n(x_{\sigma(n)}) \pi_n v_n$ in $M$, $\mathcal{L}(\mathcal{L}(q_1) \pi_1) \subseteq \mathcal{A}^*$ and $\mathcal{L}(\mathcal{L}(q_1) \pi_1) \subseteq \mathcal{A}^*$ for all $i = 1, \ldots, n$.

Then the well-formed LTW $M$ is in normal form if

- **NF1** all states are non-singleton;
- **NF2** $M$ is suffix-empty;
- **NF3** for every state $q$, if $\mathcal{L}(q) \subseteq w^*$, or $\mathcal{L}(q) \subseteq v^* u$ with $v \neq \varepsilon$, then $u = \varepsilon$;
- **NF4** for every rule $q(f(x_1, \ldots, x_n) \rightarrow v_0 q_1(x_{\sigma(1)}) \pi_1 v_1 \ldots q_n(x_{\sigma(n)}) \pi_n v_n$ in $M$, if there are $i < j$ such that $\mathcal{L}(\mathcal{L}(q_1) \pi_i v_i \ldots \mathcal{L}(q_j) \pi_j) \subseteq w^*$ and $u_i$ is a suffix of $v_{i-1}$, then $\sigma(k) < \sigma(k+1)$ for all $i \leq k < j$. 

8 Balancedness

Analogously to Lemma 18 in [3], we find:

Lemma 8. Let $M, M'$ be two LTWs that are both well-formed and in normal form. If $M$ and $M'$ are equivalent, then they are same-ordered.

In light of Lemma 7 we conclude that equivalence of $M, M'$ can be decided in polynomial time — given that they are well-formed and in normal form. Let $M$ be a well-formed LTW with productive states only. The crucial step in bringing $M$ into normal form is to achieve properties (NF2) and (NF3).

Lemma 9. Let $M$ be a well-formed LTW. Then an equivalent LTW $M'$ can be constructed in polynomial time such that for every state $q$ in $M'$,

- $\rho(L(q)) \subseteq A^*$ and
- $lcs(\rho(L(q))) = \varepsilon$.

Example 10. Consider a well-formed LTW with axiom $q(x_1)$, states $q,q'$ and the rules

\begin{align*}
q(f(x_1)) & \rightarrow abq'(x_1) \quad q'(f(x_1)) \rightarrow \overline{ab}q(x_1)ab \\
q(g) & \rightarrow ab \quad q'(g) \rightarrow ab
\end{align*}

Then $r = \varepsilon$, $r' = ab$ are the minimal words such that $rL(q)$, $r'L(q')$ are well-formed and $s = ab$ and $s' = ab$ are the longest common suffixes of $\rho(rL(q))$, $\rho(r'L(q'))$, respectively. We prepend and append $s$ and $r$ to each right-hand side of a rule of $q$ and replace each recursive call $q(x)$ by $\tau q(x)s$. We proceed with $q'$ in the same way. Thus, we obtain axiom $q(x_1)ab$ and the rules

\begin{align*}
q(f(x_1)) & \rightarrow q'(x_1) \quad q'(f(x_1)) \rightarrow q(x_1)ab \\
q(g) & \rightarrow \varepsilon \quad q'(g) \rightarrow \varepsilon
\end{align*}

The semantics did not change through the rewriting, but $\rho(L(q)), \rho(L(q')) \subseteq A^*$ and $\text{lcs}(\rho(L(q))) = \text{lcs}(\rho(L(q')))) = \varepsilon$.

Let $M$ be a well-formed LTW. As for each part $v_{i-1}L(q_i)\overline{w}$ of a rule of $M$ the longest common suffix can be computed in polynomial time, similar techniques as in the proof of Lemma 9 can be applied to obtain a suffix-empty LTW $M'$ equivalent to $M$.

Lemma 11. For a well-formed LTW $M$, an equivalent LTW $M'$ can be constructed in polynomial time that is suffix-empty.

Example 12. Let $M$ be a well-formed LTW such that for every state $q$ in $M$, $\rho(L(q)) \subseteq A^*$ and $\text{lcs}(\rho(L(q))) = \varepsilon$ hold. Let $q(f(x_1, x_2)) \rightarrow abq_1(x_1)\overline{a}q_2(x_2)\overline{p}$ be a rule of a well-formed LTW with $\varepsilon \in \rho(L(q_2))$. Then $\text{lcs}(\rho(abL(q_1)\overline{b})) = \varepsilon$ has to hold and we can rewrite the rule without changing the semantics as follows: $q(f(x_1, x_2)) \rightarrow abq_1(x_1)\overline{b}a\overline{q}_2(x_2)\overline{p}$.

Let $M$ be a well-formed LTW that is suffix-empty. Let $q$ be a state in $M$ with $\rho(L(q)) \subseteq w^*$. Then every state $q'$ reachable from $q$ is periodic, i.e., $\rho(L(q')) \subseteq \hat{w}^*$, where $\hat{w}$ is a conjugate of $w$. However, if all rules of periodic states would be in a canonical form $q(f(x_1, \ldots, x_m)) \rightarrow q_0q_1(x_{\sigma(1)}) \ldots q_n(x_{\sigma(n)})$, then $q, q_1, \ldots, q_n$ are periodic over the same period $w$ with $\rho(q_0) \in w^*$. We use this observation to eliminate all negated output symbols from right-hand sides of rules of periodic states.

Lemma 13. Let $M$ be a well-formed LTW. Then an equivalent LTW $M'$ can be constructed in polynomial time s.t. for all states $q$ of $M'$ with $\rho(L(q)) \subseteq w^*$ we have $L(q) = \rho(L(q))$, i.e., in all rules of $M'$ reachable from $q$ there occur only positive letters $a \in A$ on right-hand sides.
Example 14. Let $M$ be a well-formed LTW with states $q, q_1$ and the following rules:

\[
\begin{align*}
q(f(x)) & \rightarrow abaq_1(x)\pi \\
q_1(f(x)) & \rightarrow baq_1(x)ba \\
q_1(g) & \rightarrow \varepsilon
\end{align*}
\]

We introduce state $q_1^ab$ that has the same period $ab$ as state $q$:

\[
\begin{align*}
q(f(x)) & \rightarrow abq_1^ab(x) \\
q_1^ab(f(x)) & \rightarrow baq_1^ab(x)ba \\
q_1^ab(g) & \rightarrow \varepsilon
\end{align*}
\]

Note that we cannot remove state $q_1$ and the corresponding rules if there exists a recursive call $q_1(x)$ on a right-hand side.

Let $M$ be a well-formed LTW that is suffix-empty. Assume by Lemma 13 that for all periodic states $q$, $\rho(L(q)) = L(q) \subseteq w^*$ for some $w \in A^*$. Let $q(f(x_1, \ldots, x_m)) \rightarrow v_0q_1(x_{\sigma(1)})\overline{\pi_1}v_1 \ldots q_n(x_{\sigma(n)})\overline{\pi_n}v_n$ be a rule in $M$. If $q_1$ is a periodic state with period $w$ and $u_i \neq \varepsilon$ then we know from the suffix-empty property that $u_i \subseteq v_{i-1}$. With the techniques from the proof of Lemma 13 we can introduce a state $q_i^{w^*}$, $w' = u_iw\overline{\pi_i}$ such that $L(q_i^{w'}) = \rho(L(q_i^{w'})) = \rho(u_iL(q_i)\overline{\pi_i})$.

Lemma 15. For a well-formed LTW $M$, an equivalent LTW $M'$ can be constructed in polynomial time such that

- $M'$ is suffix-empty,
- $L(q) = \rho(L(q))$ for all states $q$ with $\rho(L(q)) \subseteq w^*$,
- for all rules $q(f(x_1, \ldots, x_m)) \rightarrow v_0q_1(x_{\sigma(1)})\overline{\pi_1}v_1 \ldots q_n(x_{\sigma(n)})\overline{\pi_n}v_n$ in $M$, if $L(q_i) \subseteq w^*$ then $u_i = \varepsilon$.

Let $M$ be well-formed LTW $M$ with the properties listed in Lemma 15. Then the next lemma shows that as a consequence also for states $q$ of $M$ with $\rho(L(q)) \subseteq w^*$, $\rho(L(q)) = L(q)$ holds.

Lemma 16. Let $M$ be a well-formed LTW that fulfills the properties listed in Lemma 15. Let $q$ be a state in $M$ with $\rho(L(q)) \subseteq vv^*$, then $L(q) = \rho(L(q))$, i.e., the output of $q$ does not contain any negated output symbols.

For a well-formed LTW $M$, thus an equivalent LTW $M'$ can be constructed in polynomial time according to Lemma 15. Then $L(q) = \rho(L(q))$ holds whenever $\rho(L(q)) \subseteq w^*$ holds where $u, v$ can be constructed in polynomial time via Theorem 31 for $\overline{\pi}$.

Lemma 17. For each well-formed LTW $M$, an equivalent LTW $M'$ can be constructed in polynomial time that is suffix-empty and for all states $q$ with $\rho(L(q)) \subseteq uv^*$ or $\rho(L(q)) \subseteq v^*u$, $v = \varepsilon$. Thus, $M'$ does not contain any ultimately periodic states that are not strictly periodic.

Given these prerequisites, we now show that every well-formed LTW $M$ can be brought into normal form in polynomial time. By Lemma 17 an equivalent LTW $M'$ can be constructed that satisfies conditions (2) and (3) of the normal form. It therefore remains to order the occurrences of periodic states in right-hand sides. Assume that $\rho(u_iL(q_i)\overline{\pi_i}v_i \ldots L(q_j)\overline{\pi_j}) \subseteq w^*$ for some suffix $u_i$ of $v_{i-1}$ holds. Then $u_iL(q_i)\overline{\pi_i}$ as well as $v_{k-1}L(q_k)\overline{\pi_k}$ are all periodic for $k = 2, \ldots, j$ with period $w$. Therefore, their ordering can be re-arranged in polynomial time according to condition (4) of the normal form.

Example 18. Let $M$ be a well-formed LTW and $q(f(x_1, x_2)) \rightarrow abq_1(x_1)\overline{b}q_2(x_1)$ be a rule in $M$ with $\rho(L(q_1)) = L(q_1) \subseteq (ab)^*$ and $\rho(L(q_2)) = L(q_2) \subseteq (ba)^*$. Then $\rho(bL(q_1)\overline{b}) \subseteq (ba)^*$ and we can rewrite the rule without changing the semantics as follows: $q(f(x_1, x_2)) \rightarrow aq_2(x_1)\overline{b}q_1(x_2)\overline{b}$. 
Altogether, we therefore have proven:

**Theorem 19.** For each well-formed LTW, an equivalent LTW can be constructed in polynomial time which is in normal form.

Let $M, M'$ be well-formed LTWs. According to Theorem 19, we may w.l.o.g. assume that both $M$ and $M'$ are in normal form. It can be checked in polynomial time whether $M$ and $M'$ are same-ordered. If they are not, $M$ and $M'$ cannot be equivalent, cf. Lemma 8. If $M$ and $M'$ are same-ordered, then their equivalence can be decided in polynomial time via reduction to the morphism equivalence problem for context-free languages and Lemma 7. In summary, we therefore obtain:

**Theorem 20.** Equivalence of well-formed LTWs is decidable in polynomial time.

Let $M$ be a 2-TW. W.l.o.g., we assume that the axiom of $M$ is of the form $q_1(x_1)q_2(x_1)$. By Proposition 6, $M$ is balanced iff both $q_1$ and $q_2$ are well-formed, and equivalent. By Theorem 21, well-formedness can be decided in polynomial time. Therefore, now assume that $q_1$ and $q_2$ are well-formed. Then we can decide the equivalence of $q_1$ and $q_2$ in polynomial time, cf. Theorem 20. This leads to our main theorem.

**Theorem 21.** Balancedness of 2-TWs is decidable in polynomial time.

4 Deciding whether a context-free language is 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 22 the maximal suffix extension of a language $L \subseteq \Sigma^*$ w.r.t. the lcs (denoted by $lcsext(L)$), i.e. the longest word $u \in \Sigma^\infty$ s.t. $lcs(uL) = u lcs(L)$. We then show that the relation $L \approx_{lcsext} L' :\iff lcs(L) = lcs(L') \land lcsext(L) = lcsext(L')$ is an equivalence relation on $\Sigma^*$ that respects both union and concatenation of languages (see Lemma 26). It then follows that for every language $L \subseteq \Sigma^*$ there is some subset $T_{lcsext}(L) \subseteq L$ of size at most 3 with $L \approx_{lcsext} T_{lcsext}(L)$.

We then use $T_{lcsext}$ to compute a finite $\approx_{lcsext}$-equivalent representation $T^{\leq h}_X$ 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 20, we show that, if $G$ is $wf$, then we have to have $T_X^{\leq 4N + 1} \approx_{lcsext} T_X^{\leq 4N}$ for all nonterminals $X$ of $G$. The complementary result is then shown in Lemma 29, i.e. if $G$ is not $wf$, then we either cannot compute up to $T_X^{\leq 4N + 1}$ as we discover some word that is not $wf$, or we have $T_X^{\leq 4N} \not\approx_{lcsext} T_X^{\leq 4N + 1}$ 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 $lcs(L)$ and $lcs(L')$, and in addition, the longest word $lcsext(L)$ resp. $lcsext(L')$ by which we can extend $lcs(L)$ resp. $lcs(L')$ when concatenating another language from left. In contrast to the computation of the lcp presented in [8], we have to take the maximal extension $lcsext$ explicitly into account. In this paragraph we do not consider the involution, thus let $\Sigma$ denote an arbitrary alphabet.

**Definition 22.** For $L \subseteq \Sigma^*$ with $R = lcs(L)$ the maximal suffix extension ($lcsext$) of $L$ is defined by $lcsext(L) := lcs(z^n \mid z \in R \in L)$. 

Note that by definition we have both \( \text{lcs}() = \text{lcs}(\emptyset) = \top \) and \( \text{lcs}({\{ R \}}) = \text{lcs}(\varepsilon^\omega) = \top \). The definition of \( \text{lcs} \) can be motivated as follows:

- **Example 23.** Consider the language \( L = \{ R, xR, yR \} \) with \( \text{lcs}(L) = R \) and \( \text{lcs}(L) = \text{lcs}(x^\omega, y^\omega) \). Assume we prepend some word \( u \in \Sigma^* \) to \( L \) resulting in the language \( uL = \{ uR, uxR, uyR \} \). Then \( \text{lcs}(uL) \) is given by \( \text{lcs}(u, \text{lcs}(L)) \text{lcs}(L) :\)

\[
\begin{align*}
\text{lcs}(u\{ xR, yR, R \}) &= \text{lcs}(u, ux, uy)R \\
&= \text{lcs}(\text{lcs}(u, ux), \text{lcs}(u, uy))R \quad \text{(as } \text{lcs}(u, ux) = \text{lcs}(u, x^\omega)) \\
&= \text{lcs}(\text{lcs}(u, x^\omega), \text{lcs}(u, y^\omega))R \\
&= \text{lcs}(u, \text{lcs}(x^\omega, y^\omega))R = \text{lcs}(u, \text{lcs}(L)) \text{lcs}(L)
\end{align*}
\]

In particular, if \( xy = yx \), we can extend \( \text{lcs} \) by any finite suffix of \( \text{lcs}(L) = (xy)^\omega \) — note that, if \( x = \varepsilon = y \), then \( \text{lcs}(L) = \top \) by our definition that \( \varepsilon^\omega = \top \); but if \( xy \neq yx \), then we can extend it at most to \( \text{lcs}(L) = \text{lcs}(x^\omega, y^\omega) = \text{lcs}(xy, yx) \sqsubseteq xy \).

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

- **Lemma 24.** Let \( L \subseteq \Sigma^* \) with \( |L| \geq 2 \) and \( R := \text{lcs}(L) \). Fix any \( xR \in L \setminus \{ R \} \). Then there is some \( y \in L \setminus \{ R \} \) s.t. \( \text{lcs}(L) = \text{lcs}(x^\omega, y^\omega) = \text{lcs}(x^\omega, y^\omega, z^\omega) \) for all \( zR \in L \). If \( xy = yx \), then \( R \in L \).

We show that we can compute the \( \text{lcs} \) and the extension \( \text{lcs} \) of the union resp. the concatenation of two languages solely from their \( \text{lcs} \) and \( \text{lcs} \). To this end, we define the \( \text{lcs} \)-summary of a language as:

- **Definition 25.** For \( L \subseteq \Sigma^* \) set \( \text{lcssum}(L) := (\text{lcs}(L), \text{lcs}(L)) \). The equivalence relation \( \approx_{\text{lcs}} \) on \( 2^{\Sigma^*} \) is defined by: \( L \approx_{\text{lcs}} L' \) iff \( \text{lcssum}(L) = \text{lcssum}(L') \).

- **Lemma 26.** Let \( L, L' \subseteq \Sigma^* \) with \( \text{lcssum}(L) = (R, E) \) and \( \text{lcssum}(L') = (R', E') \). If \( L = \emptyset \) or \( L' = \emptyset \), then \( \text{lcssum}(L \cup L') = (\text{lcs}(R, R'), \text{lcs}(E, E')) \), and \( \text{lcssum}(LL') = (\top, \top) \). Assume thus \( L \neq \emptyset \neq L' \) which implies \( R \neq \top \neq R' \). Then:

\[
\begin{align*}
\text{lcs}(L \cup L') &= \text{lcs}(R, R') \quad \text{and} \quad \text{lcs}(LL') = \text{lcs}(R, E')R'. \\
\text{If } \text{lcs}(R, R') \notin \{ R, R' \}, \text{ then } \text{lcssum}(L \cup L') = \varepsilon \quad \text{else w.l.o.g. } R' = \delta R \text{ and } \text{lcssum}(L \cup L') = \text{lcssum}(E, \text{lcs}(E', \delta)) \delta . \\
\text{If } \text{lcs}(R, E') \subseteq R, \text{ then } \text{lcssum}(LL') = \varepsilon \quad \text{else } E' = \delta R \text{ and } \text{lcssum}(LL') = \text{lcssum}(E, \delta ).
\end{align*}
\]

- **Example 27.** Consider \( L = \{ a, ba \} \) and \( L' = \{ aa, baaa \} \) s.t. \( \text{lcssum}(L) = (a, (ba)^\omega) \) and \( \text{lcssum}(L') = (aa, (ba)^\omega) \). Then \( \text{lcs}(L \cup L') = \text{lcs}(a, a) = a, \text{lcs}(LL') = \text{lcs}((ba)^\omega, aa = a a a, \text{lcs}(L \cup L') = \text{lcs}((ba)^\omega, ((ba)^\omega, (ba)^\omega a) = a, \text{ and } \text{lcs}(LL') = \text{lcs}((ba)^\omega, (ab)^\omega) = \varepsilon \) as \( (ba)^\omega = (ab)^\omega a \).

As both the \( \text{lcs} \) and the \( \text{lcs} \) are determined by already two words (cf. Lemmas 2 and 24), it follows that every \( L \subseteq \Sigma^* \) is \( \approx_{\text{lcs}} \)-equivalent to some sublanguage \( T_{\text{lcs}}(L) \subseteq L \) consisting of at most three words where the words \( xR, yR \) can be chosen arbitrarily up to the stated constraints:

\[
T_{\text{lcs}}(L) := \begin{cases}
L & \text{if } |L| \leq 2 \\
\{ R, xR, yR \} & \text{if } \{ R, xR, yR \} \subseteq L \wedge \text{lcs}(L) = \text{lcs}(x^\omega, y^\omega) \quad \text{with } R := \text{lcs}(L) \\
\{ xR, yR \} & \text{if } R = \text{lcs}(xR, yR) \wedge R \notin L \wedge \{ xR, yR \} \subseteq L.
\end{cases}
\]
Deciding well-formedness

For the following, we assume that \( G \) is a context-free grammar over \( B = A \cup \overline{A} \) with nonterminals \( X \). Set \( N := \{X\} \). We further assume that \( G \) is nonnegative, and that we have computed for every nonterminal \( X \) of \( G \) a word \( r_X \in \mathbb{A}^* \) (represented as an SLP) s.t. \( |r_X| = d_X \) and \( r_X \in \text{Prf}(\rho(L_X)) \).\(^2\) In order to decide whether \( G \) is \( \text{wf} \) we compute the languages \( \rho(r_X L_X^{<h}) \) modulo \( \text{ics} \) for increasing derivation height \( h \) using fixed-point iteration. Assuming inductively that (i) \( r_X L_X^{<h} \) is \( \text{wf} \) and that we have computed (ii) \( T_X^{<h} := \text{ics}(\rho(r_X L_X^{<h})) \approx_{\text{ics}} \rho(r_X L_X^{<h}) \) for all \( X \in X \) up to height \( h \) we can compute \( \text{ics}(\rho(r_X L_X^{<h+1})) \) for each nonterminal as follows:

\[
\rho(r_X L_X^{<h+1}) = \rho(r_X L_X^{<h}) \cup \bigcup_{Y \rightarrow aY} \rho(r_Y L_Y^{<h}) \cup \bigcup_{X \rightarrow \gamma YZ} \rho(r_X L_X^{<h} T_Y^{<h} r_Z L_Z^{<h}) \\
\approx_{\text{ics}} T_X^{<h} \cup \bigcup_{Y \rightarrow aY} \rho(r_Y L_Y^{<h} T_Y^{<h}/r_Z L_Z^{<h}) \\
\approx_{\text{ics}} \text{ics}(\rho(T_X^{<h} \cup \bigcup_{Y \rightarrow aY} r_X L_X^{<h} T_Y^{<h} \cup \bigcup_{X \rightarrow \gamma YZ} r_X L_X^{<h} r_Z L_Z^{<h} ) ) \\
= : T_X^{<h+1}
\]

Note that, if all constants \( r_X \) and all \( T_X^{<h} \) are \( \text{wf} \), but \( G \) is not \( \text{wf} \), then the computation has to fail while computing \( r_X L_X^{<h} T_X^{<h} r_Z L_Z^{<h} \). See the following Example.

**Example 28.** Consider the nonnegative grammar \( G \) given by the rules (with \( n \in \mathbb{N} \) fixed)

\[
\begin{align*}
S & \rightarrow Uc \quad U \rightarrow AV | W_3 \\
A & \rightarrow a \quad B \rightarrow b \quad \overline{B} \rightarrow \overline{b} \\
W_1 & \rightarrow W_3 \\
W_2 & \rightarrow W_1 W_1 \\
W_3 & \rightarrow BB
\end{align*}
\]

with axiom \( S \). Except for \( \overline{B} \) all nonterminals generate nonnegative languages. Note that the nonterminals \( W_1 \) to \( W_3 \) form an SLP that encodes the word \( b^{2n} \) by means of iterated squaring which only becomes productive at height \( h = n + 1 \). For \( h \geq n + 1 \) we have:

\[
\begin{align*}
L_{X}^{<h} & = \{a^{k}b^{2^{n-k}}c | k \leq \frac{h-(n+3)}{2}\} \\
L_{Y}^{<h} & = \{a^{k}b^{2^{n-k}}c | k \leq \frac{h-(n+3)}{2}\} \\
L_{Z}^{<h} & = \{a^{k}b^{2^{n-k}}c | k \leq \frac{h-(n+3)}{2}\} \\
L_{A}^{<h} & = \{a\} \\
L_{B}^{<h} & = \{\overline{b}\}
\end{align*}
\]

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 \( r_X L_X^{<h} \) is \( \text{wf} \) for all nonterminals \( X \) up to \( h \leq h_0 = 2^{n+2} + (n+2) \) s.t. \( \text{ics}(\rho(r_S L_S^{<h})) \approx_{\text{ics}} T_S^{<h} = \{b^{2n}c, a^{k}b^{2^{n-k}}c\} \) for \( k = \max(0, ((h - (n+3))/2)) \) and \( n + 1 \leq h \leq h_0 \); in particular, the \( \text{ics} \) of \( T_S^{<h} \) converges immediately to \( c \), only its maximal extension \( \text{ics} \) changes for \( n + 1 \leq h \leq h_0 \). We discover the first counterexample \( a^{2^{n}}\overline{b} \) that \( G \) is not \( \text{wf} \) while computing \( T_{\overline{b}}^{<h+1} = \text{ics}(\rho(T_{\overline{b}}^{<h+1})) \).

As illustrated in Example 28, if \( G \) is not \( \text{wf} \), 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 \( T_X^{<h} \) cannot have converged (modulo \( \text{ics} \)).

**Lemma 29.** If \( L = L(G) \) is not \( \text{wf} \), then there is some least \( h_0 \) s.t. \( r_X L_X^{<h} T_X^{<h} \) is not \( \text{wf} \) with \( X \rightarrow G YZ \). For \( h \leq h_0 \), all \( r_X L_X^{<h} \) are \( \text{wf} \) s.t. \( T_X^{<h} \approx_{\text{ics}} \rho(r_X L_X^{<h}) \). If \( h_0 \geq 4N + 1 \), then at least for one nonterminal \( X \) we have \( T_X^{<h+1} \not\approx_{\text{ics}} T_X^{<h+N} \).

\(^2\) \( r_X \) is (after reduction) a longest word of closing brackets in \( \rho(L_X) \) (if \( G \) is \( \text{wf} \), then \( r_X \) is unique). An SLP encoding \( r_X \) can be computed in polynomial time while checking that \( G \) is nonnegative; see Definition 4 and the subsequent explanations, and the proof of Lemma 8 in the appendix of the extended version.
The following Lemma \[30\] states the complementary result, i.e. if \( G \) is \( \text{wf} \) then the representations \( T_X^h \) have converged at the latest for \( h = 4N \) modulo \( \text{lcs} \). The basic idea underlying the proof of Lemma \[30\] is similar to [8]; 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 \( \text{lcs} \) (after reduction). In contrast to [8] we need not only to show that \( T_X^h \) has the same \( \text{lcs} \) as \( \rho(r_X L_X^h) \), but that \( T_X^h \) has converged modulo \( \text{lcs} \) if \( G \) is \( \text{wf} \); to this end, we need to explicitly consider \( \text{lcs} \), and re-prove stronger versions of the results regarding the combinatorics on words which take the involution into account (see Section A.12 in the appendix of the extended version) \[1\].

\[ \textbf{Lemma 30.} \] let \( G \) be a context-free grammar with \( N \) nonterminals and \( L := L(G) \) \( \text{wf} \). For every nonterminal \( X \) let \( r_X \in A^* \) s.t. \( |r_X| = d_X \) and \( r_X L_X \) \( \text{wf} \). Then \( \rho(r_X L_X) \approx_{\text{lcs}} \rho(r_X L_X^h) \), and thus \( T_X^{h+1} \approx_{\text{lcs}} T_X^h \) for every nonterminal \( X \).

As \( |T_X^h| \leq 3 \), a straight-forward induction also shows that every word in \( T_X^h \) 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 \[29\] and \[30\] we thus obtain the main result of this section:

\[ \textbf{Theorem 31.} \] Given a context-free grammar \( G \) over \( B \) we can decide in time polynomial in the size of \( G \) whether \( G \) is \( \text{wf} \).

5 Conclusion

We have shown that well-formedness for context-free languages is decidable in polynomial time. We have also presented a polynomial-time algorithm for deciding equivalence of well-formed LTWs. This allowed us to decide in polynomial time whether or not a 2-TW is balanced. The question remains whether balancedness is decidable also for more general MSO definable transductions. It is also open whether even the single bracket case can be generalized beyond MSO definable transduction, e.g., to the output languages of topdown tree-to-word transducers \[13\].

References

1. J. Berstel and L. Boasson. Formal properties of XML grammars and languages. Acta Inf., 38(9):649–671, 2002.
2. Adrien Boiret. Normal form on linear tree-to-word transducers. In Adrian-Horia Dediu, Jan Janoušek, Carlos Martín-Vide, and Bianca Truthe, editors, Language and Automata Theory and Applications: 10th International Conference, LATA 2016, pages 439–451, LNCS 9618, Springer, 2016. URL: http://dx.doi.org/10.1007/978-3-319-30000-9_34
3. What prevents us to apply the results of [8] to the reduced \( \text{lcs} \) is, roughly spoken, that given a well-formed linear grammar of the following form

\[ S \rightarrow uX \quad X \rightarrow s_1X\overline{T_1}t_1r_1 \mid s_2X\overline{T_2}t_2r_2 \mid s_3X\overline{T_3}t_3r_3 \mid w \quad (u, w, s_i, t_i, r_i \in A^*) \]

we cannot in general find suitable conjugates of \( u, w, s_i, t_i, r_i \) that allow us to cancel the factors \( \overline{T_i} \) in each rule while preserving the structure of the grammar and its language after reduction; see Example \[46\] in the appendix. Still, we can proceed as in Lemma \[15\] to remove closing brackets for all nonterminals that produce an ultimately periodic language after reduction; although, we currently do not know if we can transform in polynomial time a well-formed context-free grammar \( G \) over \( A \) s.t. \( \rho(L(G)) = L(G') \) (and further s.t. derivations are in bijection).
A Appendix

Contents

A.1 Lemma 7 in the main work ........................................ 15
A.2 Lemma 9 in the main work ........................................ 15
A.3 Lemma 11 in the main work ....................................... 16
A.4 Lemma 13 in the main work ....................................... 16
A.5 Lemma 16 in the main work ....................................... 17
A.6 Lemma 17 in the main work ....................................... 17
A.7 Properties of the longest common prefix and suffix ........ 18
A.8 Lemma 24 in the main work ....................................... 22
A.9 Lemma 26 in the main work ....................................... 23
A.10 Lemma 4 in the main work ....................................... 26
A.11 Lemma 5 in the main work ....................................... 27
A.12 Reduced LCS of simple linear \(wf\) languages ............. 29
A.13 Lemma 30 in the main work ..................................... 42
A.14 Lemma 29 in the main work ..................................... 48

A.1 Lemma 7 in the main work

\textbf{Lemma.} The morphism equivalence problem of a context-free grammar in a free group is decidable in polynomial time.

\textbf{Proof.} By Plandowski [11, Theorem 6], showed for finitely generated free groups, that a polynomial size test set can be constructed from a context-free grammar in polynomial time. Thereby, the words in the test set are represented by SLPs. Equivalence of two SLPs over a free group, on the other hand, has been shown to be decidable in polynomial time by Lohrey [6]. Together, therefore, the statement of the lemma follows. \hfill \square

A.2 Lemma 9 in the main work

\textbf{Lemma 32.} Let \(M\) be a well-formed LTW. Then an equivalent LTW \(M'\) can be constructed in polynomial time such that for every state \(q\) in \(M'\),
\begin{itemize}
  \item \(\rho(\mathcal{L}(q)) \subseteq A^*\) and
  \item \(\text{lcs}(\rho(\mathcal{L}(q))) = \varepsilon\).
\end{itemize}

\textbf{Proof.} Let \(q\) be a state of a well-formed LTW \(M\). Then \(\mathcal{L}(q)\) is bounded well-formed and (an SLP for) the minimal word \(r_q \in A^*\) can be computed such that \(r_q \mathcal{L}(q)\) is \(wf\) and \(s_q := \text{lcs}(r_q \mathcal{L}(q))\), cf. Section 4. For every rule \(q(f(x_1, \ldots, x_m)) \rightarrow \gamma q_1(x_{\sigma(1)}) \gamma_1 \ldots q_n(x_{\sigma(n)}) \gamma_n\) we obtain a rule
\[q'(f(x_1, \ldots, x_m)) \rightarrow r_q \gamma r_q r_q r_q'(x_{\sigma(1)}) s_q r_q s_q \gamma_1 \gamma_2 \ldots r_q r_q r_q r_q r_q'(x_{\sigma(n)}) s_q r_q s_q \gamma_n \gamma_q \]
Let \(ax = \gamma_0 q(x_1) \gamma_1\) be the axiom in \(M\), then we add the axiom \(ax' = \gamma_0 r_q q'(x_1) s_q \gamma_1\) to \(M'\).

Let \(q\) be a state in \(M\). We prove by induction over the size of the input tree that for all \(t \in T_\Sigma\), \([q'](t) = r_q [q'](t) \overline{s_q} \). For the base case let \(t = h \in \Sigma(0)\) and \(q(h) \rightarrow \gamma_0\) be the corresponding rule in \(M\). Then \([q'](h) = r_q \gamma_0 \overline{s_q} = r_q [q'](h) \overline{s_q} \). Let \(f(x_1, \ldots, x_n) \in \Sigma^*\) be an input tree in \(T_\Sigma\) for \(\mathcal{L}(q)\). By the inductive hypothesis, \([q'](f(x_1, \ldots, x_n)) = r_q [q'](f(x_1, \ldots, x_n)) \overline{s_q}\). For the inductive case let \(q\) be a rule of \(M\) and \(q'(x_1, \ldots, x_m) \rightarrow r_q q_1(x_{\sigma(1)}) q_1' \ldots q_n(x_{\sigma(n)}) q_n'\). Then \([q'](f(x_1, \ldots, x_n)) = r_q q_1(x_{\sigma(1)}) q_1'(f(x_{\sigma(1)})) \ldots q_n(x_{\sigma(n)}) q_n'(f(x_{\sigma(n)})) \overline{s_q}\). Finally, let \(q\) be a rule of \(M'\) and \(q'(x_1, \ldots, x_m) \rightarrow r_q q_1(x_{\sigma(1)}) q_1' \ldots q_n(x_{\sigma(n)}) q_n'\). Then \([q'](f(x_1, \ldots, x_n)) = r_q q_1(x_{\sigma(1)}) q_1'(f(x_{\sigma(1)})) \ldots q_n(x_{\sigma(n)}) q_n'(f(x_{\sigma(n)})) \overline{s_q}\). Together, therefore, the statement of the lemma follows.
balancedness

$\mathcal{T}_\Sigma^{k+1}$ and assume that for all $t \in \mathcal{T}_\Sigma^{\leq k+1}$, $[q'](t) = r_q[q](t)^{\overline{\gamma_q}}$. Let $q(f(x_1, \ldots, x_m)) \rightarrow \gamma_0q_1(x_{\sigma(1)}))^\gamma_1 \ldots q_n(x_{\sigma(n)})^\gamma_n$ be the corresponding rule in $M$. Then

$$[q'](f(x_1, \ldots, x_m)) = r_q[r_q[q_1](x_{\sigma(1)})^s_1 \gamma_1 \ldots r_q[q_n](x_{\sigma(n)})^s_n]^\gamma_n \overline{\gamma_q},$$

$$= r_q[r_q[q_1](x_{\sigma(1)})^s_1 \gamma_1 \ldots r_q[q_n](x_{\sigma(n)})^s_n]^\gamma_n \overline{\gamma_q},$$

$$= r_q[r_q[q_1](x_{\sigma(1)})^s_1 \gamma_1 \ldots r_q[q_n](x_{\sigma(n)})^s_n]^\gamma_n \overline{\gamma_q},$$

$$= r_q[r_q[q_1](x_{\sigma(1)})^s_1 \gamma_1 \ldots r_q[q_n](x_{\sigma(n)})^s_n]^\gamma_n \overline{\gamma_q}.$$

Let $ax = \gamma_0q(x_1)^\gamma_1$ be the axiom in $M$. Then for all $t \in \mathcal{T}_\Sigma$, $[ax'] = r_q[r_q[q](t)](t)^{\overline{\gamma_q}} = \gamma_0[q](t)^{\overline{\gamma_q}} = \gamma_0[q](t)^{\overline{\gamma_q}} = [ax]$. Thus, $M$ and $M'$ are equivalent. From the construction it directly follows that for all states $q$ in $M'$, $\rho(\mathcal{L}(q)) \subseteq A^*$ and $\text{lcs}(\rho(\mathcal{L}(q))) = \varepsilon$.

### A.3 Lemma 11 in the main work

**Lemma.** For a well-formed LTW $M$, an equivalent LTW $M'$ can be constructed in polynomial time that is suffix-empty.

**Proof.** W.l.o.g. we assume by Lemma that all states in $M$ are already well-formed, i.e. $\rho(\mathcal{L}(q)) \subseteq A^*$, and that $\text{lcs}(\rho(\mathcal{L}(q))) = \varepsilon$. We start with a copy of $M$ for $M'$. For each rule $q(f(x_1, \ldots, x_m)) \rightarrow v_0q_1(x_{\sigma(1)})^m_1 \ldots q_n(x_{\sigma(n)})^m_n$ in $M$ we rewrite the rule iteratively from $i = 1, \ldots, n$ as follows. Let $s_i := \text{lcs}(\rho(v_{i-1}\mathcal{L}(q_i)^m_i))$, then we let

$$q(f(x_1, \ldots, x_m)) \rightarrow v_0q_1(x_{\sigma(1)})^m_1 \ldots q_i(x_{\sigma(i)})^m_i \ldots q_n(x_{\sigma(n)})^m_n$$

From the construction follows that after the $i$-th iteration for all $1 \leq j \leq i$,

$$\text{lcs}(\rho(v_{i-1}\mathcal{L}(q_i)^m_i)) = \varepsilon$$

Thus, after all iterations $M'$ is suffix-empty while the rewriting did not change the semantics as

$$v_0[q_1](t_1)^{\overline{m}_1} \overline{s}_1 v_1 \ldots v_n[q](t)^{\overline{m}_n} \overline{s}_n v_n \rightarrow v_0(q_1)(t)^{\overline{m}_i} v_1 \ldots v_n[q](t)^{\overline{m}_n} \overline{s}_n v_n$$

for all $t_1, \ldots, t_i \in \mathcal{T}_\Sigma$ and $i = 1, \ldots, n$. With the suffix representation of each $\mathcal{L}(q_i)$ the words $s_i$ can be computed in polynomial time and therefore the overall rewriting runs in polynomial time.

### A.4 Lemma 13 in the main work

**Lemma.** Let $M$ be a well-formed LTW. Then an equivalent LTW $M'$ can be constructed in polynomial time s.t. for all states $q$ of $M$ with $\rho(\mathcal{L}(q)) \subseteq w^*$ we have $\mathcal{L}(q) = \rho(\mathcal{L}(q))$, i.e., in all rules of $M'$ reachable from $q$ there occur only positive letters $a \in A$ on right-hand sides.

**Proof.** W.l.o.g. we assume that $M$ is suffix-empty. Let

$$q(f(x_1, \ldots, x_m)) \rightarrow v_0q_1(x_{\sigma(1)})^m_1 \ldots q_n(x_{\sigma(n)})^m_n$$

be a rule in $M$ with $\rho(\mathcal{L}(q)) \subseteq w^*$. If $u_i \neq \varepsilon$ then $u_i \subseteq v_{i-1} = v_{i-1}u_i$ as $M$ is suffix-empty. Thus

$$[q](f(x_1, \ldots, x_m)) = v_0^m T_{\mathcal{L}(q)}^{u_i} x_{\sigma(1)}^{v_i} \ldots x_{\sigma(n)}^{v_n}$$

with $T_{\mathcal{L}(q)}^{u_i}(t) = u_i(q)(t)^{\overline{m}_i}$. Let $w_i$ be a reduced output word of $q_i$, i.e., $w_i \in \rho(\mathcal{L}(q_i))$ and $w_i = u_i[w_i^m_i]$ the corresponding conjugate. Then, for all $\hat{w} \in \rho(u_i\mathcal{L}(q_i)^m_i)$, $j = 1, \ldots, n$,

$$v_i^j w_i^j v_i^j \ldots v_i^j \hat{w} v_i^j \ldots w_n^j \subseteq w^*.$$
Thus, the languages $\rho(uL(q_1)\overline{w})$ and $\rho(L(q))$ are periodic over some conjugate of $w$. As $\text{lcs}(\rho(L(q_1))) = \varepsilon$, we know that $\varepsilon \in \rho(L(q_1))$. Thus, for a rule $q'(f(x_1, \ldots, x_n)) \rightarrow \gamma_0 q_1(x_{\sigma(1)}) \gamma_1 \ldots q_n(x_{\sigma(n)}) \gamma_n$ reachable from a periodic state $q$ with $\rho(L(q)) \subseteq w^*$ we have

- $\rho(\gamma_0 \ldots \gamma_n) \subseteq (w''w')^*$ with $w = w''w''$,
- $\rho(L(q_1)) \subseteq (\hat{w}w)^*$ with $w = \hat{w}w$,
- $\varepsilon \in \rho(L(q_1))$

Thus, if every rule reachable from $q$ with $\rho(L(q)) \subseteq w^*$ has the form $q'(f(x_1, \ldots, x_n)) \rightarrow w_0 q_1(x_{\sigma(1)}) \ldots q_n(x_{\sigma(n)}) \gamma$ then $w_0 \in w^*$ and $\rho(L(q_1)) \subseteq w^*$.

We base our construction on the above observations. Let $M'$ be a copy of $M$ and $q$ be a periodic state in $M'$ with $\rho(L(q)) \subseteq w^*$. For each state $\hat{q}$ reachable from $q$ with rule $\hat{q}(f(x_1, \ldots, x_n)) \rightarrow \gamma_0 q_1(x_{\sigma(1)}) \gamma_1 \ldots q_n(x_{\sigma(n)}) \gamma_n$ we add the rule

$$\hat{q}^w(f(x_1, \ldots, x_n)) \rightarrow w^k q_1^w(x_{\sigma(1)}) \ldots q_n^w(x_{\sigma(n)})$$

with $k = |\rho(\gamma_0 \ldots \gamma_n)|/|w|$ to $M'$. We replace every recursive call $q(x)$ on the right-hand side of a rule by $q^w(x)$ and remove all rules for $q$. With the above observations of the periodicity of the states we can inductively show that $[q](t) = \hat{[q]}(t)$ for all $t \in T_S$. Let $k$ be the maximal number of rules reachable from a state $q$ and $\ell$ be the number of periodic states in $M$. Then the size of the transducer is increased by at most $\ell \cdot k$ rules – a polynomial size increase. Note that the size increase may be less if the period $w$ of an introduced state $q^w$ is the same as the period of state $q$.

### A.5 Lemma 16 in the main work

**Lemma.** Let $M$ be a well-formed LTW that fulfills the properties listed in Lemma 15. Let $q$ be a state in $M$ with $\rho(L(q)) \subseteq uu^*$, then $L(q) = \rho(L(q))$, i.e., the output of $q$ does not contain any negated output symbols.

**Proof.** W.l.o.g. we assume that $M$ does only contain nonsingleton states, i.e., for $q$ a state in $M$, $\rho(L(q))$ does contain at least two words. Let $q$ be a state in $M$ such that $\rho(L(q)) = uu^*$. As $M$ is suffix-empty, we know that $\text{lcs}(u,v) = \varepsilon$ and $\text{lec}(\rho(L(q))) = u$. Let $q(f(x_1, \ldots, x_n)) \rightarrow v_0 q_1(x_{\sigma(1)}) v_1 \ldots q_n(x_{\sigma(n)}) v_n$ be a rule for $q$. Then either

- $v_0 \not\subseteq u = v_0 u'$, $\rho(L(q_1)) \subseteq u'v^*$ and $\rho(v_1 L(q_1) v_1 \ldots L(q_n) v_n v_n) \subseteq v^*$ or
- $u \not\subseteq v_0 = wv_0, \rho(v_0 L(q_1) v_1 \ldots L(q_n) v_n v_n) \subseteq v^*$.

In the first case $u_1$ has to be empty as $\text{lcs}(u,v) = \varepsilon$ and $u_2, \ldots, u_n$ are empty as $L(q_i), i = 2, \ldots, n$ are periodic and the conditions of Lemma 15 hold. In the second case $u_1, \ldots, u_n$ are empty as all $L(q_i), i = 1, \ldots, n$ are periodic. Therefore all $u_i$ have to be empty and $L(q) = \rho(L(q))$ if $\rho(L(q)) \subseteq uu^*$.

### A.6 Lemma 17 in the main work

**Lemma.** For well-formed LTW $M$, an equivalent LTW $M'$ can be constructed in polynomial time that is suffix-empty and for all states $q$ with $\rho(L(q)) \subseteq uu^*$ or $\rho(L(q)) \subseteq v^*u$, $v = \varepsilon$. Thus, $M'$ does not contain any ultimately periodic states that are not strictly periodic.

**Proof.** Let $M'$ be an equivalent LTW such that the conditions of Lemma 15 hold. Thus, $M'$ is suffix-empty, for all states $q$ in $M$ that produce a periodic language after reduction, $\rho(L(q)) = L(q)$, and no recursive call of such a periodic state is followed by a word $\overline{w}$. As $M'$ is suffix-empty there are no states $q$ with $\rho(L(q)) \subseteq v^*u$ with $u \neq \varepsilon$. Thus, we only have to consider states $q$ with $\rho(L(q)) \subseteq uu^*$ with $u, v \neq \varepsilon$ and $\text{lcs}(u,v)\varepsilon$. Let $q$ be a state in $M'$
Balancedness

In polynomial time. Similar to the approach in [3] we show how we can construct \( q_0^q \) for \( q \) with \( \mathcal{L}(q) \subseteq \mathbb{w}^* \) such that \( [q_0^q](t) = u[q^q](t), t \in T_\Sigma \).

Let \( q \) be a state in \( M' \) with \( \mathcal{L}(q) \subseteq \mathbb{w}^* \). Then \( u = \text{lcp}(\rho(\mathcal{L}(q))) \) as \( M' \) is suffix-empty. Let \( q(f(x_1, \ldots, x_m) \rightarrow v_0q_1(x_{\sigma(1)})v_1 \cdots q_n(x_{\sigma(n)})v_n) \) be a rule for \( q \). W.l.o.g. we assume that \( M' \) does contain only non-singleton states. Thus, \( \rho(\mathcal{L}(q_1)) \) contains at least two words. Therefore \( \rho(\mathcal{L}(q_1)) \) is either ultimately periodic of the form \( v_0^q v^* \) with \( u \sqsubseteq v_0 = uv_0^* \) or \( \rho(\mathcal{L}(q_1)) \) is periodic and \( u \sqsubseteq v_0 \). This leads to the following construction.

**Case 1**: If \( u \text{lcp} v_0 = vr_0^q \) then we know that all states \( q_i \) are periodic. Therefore, we add a rule

\[
q^q(f(x_1, \ldots, x_n)) \rightarrow v_0^q q_1(x_{\sigma(1)})v_1 \cdots q_n(x_{\sigma(n)})v_n
\]

to \( M' \).

**Case 2**: If \( u = v_0r_0^q \) then \( \rho(\mathcal{L}(q_1)) = \mathcal{L}(q_1) \subseteq u^wv^* \) as \( \text{lcs}(\mathcal{L}(q_1)) = \varepsilon \) and \( v_1 \mathcal{L}(q_2)v_2 \cdots \mathcal{L}(q_n)v_n \subseteq v^* \). Thus, we add a rule

\[
q^q(f(x_1, \ldots, x_n)) \rightarrow q_1^q(x_{\sigma(1)})v_1 q_2(x_{\sigma(2)}) \cdots q_n(x_{\sigma(n)})v_n
\]
to \( M' \). State \( q_1^q \) is constructed in the same way.

For each rule \( q \) for which we constructed \( q^q \) we can remove all rules for \( q \) and replace every recursive call \( q(x) \) on a right-hand side of a rule or in the axiom of \( M' \) by \( w^q(x) \). With the observations above, we can inductively show that \( u[q^q](t) = [q](t) \) and therefore the semantics of \( M' \) does not change and is still equivalent to \( M \).

The remaining part of the appendix is self-contained, i.e. we restate all definitions and lemmata of the main work, give the missing proofs, and also introduce further lemmata needed to prove the main results.

### A.7 Properties of the longest common prefix and suffix

Fix some generic nonempty finite alphabet \( \Sigma \) with \( \top \not\in \Sigma \) a fresh, unused symbol. We write \( \Sigma^\infty \) for \( \{ \top \} \cup \Sigma^* \cup \Sigma^\omega \). As mentioned in the main work, the \( \text{lcp} \) is the infimum w.r.t. the prefix order \( \sqsubseteq \) on \( \Sigma^\infty \) extended by a greatest element \( \top \) in order to handle the empty set. We briefly sketch the argument: For \( u \in \Sigma^\infty \), set \( h(u) := u\Sigma^\infty \), if \( u \in \Sigma^* \), and \( h(u) := \{ u \} \) otherwise. We then have \( h(u) = \{ w \in \Sigma^\omega \mid u \sqsubseteq w \} \), i.e. we can alternatively define \( \sqsubseteq \) by means of \( u \sqsubseteq v \iff h(u) \supseteq h(v) \). Thus, \( \top \) becomes the greatest element w.r.t. \( \sqsubseteq \) by setting \( h(\top) := \emptyset \). Extend \( h \) to languages \( L \subseteq \Sigma^\infty \) by means of \( h(L) := \bigcup_{w \in L} h(w) \). The \( \text{lcp}(L) \) is then the unique word in \( \Sigma^\infty \) satisfying \( h(\text{lcp}(L)) = \hat{h}(\{ \text{lcp}(L) \}) = \bigcap\{ z \Sigma^\omega \mid \hat{h}(L) \subseteq z \Sigma^\omega \} \), and thus is the infimum w.r.t. \( \sqsubseteq \).

In the following, we summarize some properties of the \( \text{lcs} \) which are used in the following proofs. For easier reference, we restate the definition of the \( \text{lcs} \) and \( \Sigma^{\text{ulp}} \):

**Definition 33** (Definition 1 in the main work).

- For \( u \in \Sigma^* \) and \( w \in \Sigma^+ \), define the expression \( w^\omega u \) by means of \( w^\omega u := (u R^R) w^\omega R^R \), and its reverse by means of \( (w^\omega u)^R = (w R^R)^\omega u R^R \). The set of ultimately left-periodic words is then \( \Sigma^{\text{ulp}} := \{ w^\omega u \mid w \in \Sigma^+, u \in \Sigma^* \} \).
- The suffix order on \( \Sigma^* \cup \Sigma^{\text{ulp}} \cup \{ \top \} \) is then defined by \( u \sqsubseteq v \iff u R^R \sqsubseteq v R^R \).
- The longest common suffix (lcs) of a language \( L \subseteq \Sigma^* \cup \Sigma^{\text{ulp}} \) is then \( \text{lcs}(L) := \text{lcp}(L R^R) R^R \).
Note that for the lcs a term like \( \text{lcs}(x^n, \ldots) \) is always supposed to be read as \( \text{lcs}(\ldots xxxxx, \ldots) = \text{lcp}(x^R)^\omega, \ldots)^R \).

We prove the following properties of the lcs which allow to simplify the computation of the lcs, in particular in the case of ultimately periodic words.

\[ \text{Lemma 34.} \quad \text{Let } u, v, w, x, y, z \in \Sigma^*. \]

\[ u^n v = \begin{cases} u^n u^k v & (\forall k \geq 0) \\ u \subseteq x^ny \quad \text{iff} \quad x = \varepsilon \lor \exists k: u \subseteq x^k y \\ u \subseteq w^n \subseteq \top \quad \text{iff} \quad u \subseteq uw \\ u \subseteq w^n \quad \text{iff} \quad \exists w', w'' , k: w = w'w'' \land w'' \subseteq w \land u = w''u^k \\ u \subseteq w^n \quad \text{iff} \quad u \subseteq uw \\ u^n v = w^n \quad \text{iff} \quad \exists p, q \in \Sigma^*: pv = vq \land u \in p^+ \land w \in q^* \\ u^n v = x^n y \quad \text{iff} \quad u = \varepsilon = x \lor u \neq \varepsilon \neq x \land \forall k \exists !: u^k v \subseteq x^1y \land x^ky \subseteq u^1v \\ u^n v = xy x \quad \text{iff} \quad u = \varepsilon = x \lor u \neq \varepsilon \neq x \land \forall k, l: u^k u^l \omega \text{ is weakly well-formed} \\ \text{lcs}(x, xy) = \begin{cases} y^n \quad \text{if } x = \varepsilon \\ \text{lcs}(x, y^n) \end{cases} \\ \text{lcs}(x^n, y^n) = \begin{cases} y^n \quad \text{if } xy = yx \land x \neq \varepsilon \\ \text{lcs}(x^n, y^n) = \text{lcs}(x^n, y^n) = \text{lcs}(x^n, y^n) = \text{lcs}(x^n, y^n) \end{cases} \\ \text{lcs}(u^n, w^n) = \begin{cases} \text{lcs}(u^n, w^n)u \end{cases} \]

\[ \text{Proof.} \]

1. \( \forall k: u^n v = u^n u^k v \) as
   \( u = \varepsilon \), then:
   \( u^n v = \top v = \top = \top u^n v = u^n u^k v \) for all \( k \).
   So assume \( u \neq \varepsilon \).
   Then by definition for any \( k \in \mathbb{N}_0: \)
   \[ u^n v = u^n u^k v \text{ iff } \bigcap_{n \geq b} u^R(u^R)^n \Sigma^\omega = \bigcap_{n \geq k} u^R(u^R)^n \Sigma^\omega \]

2. \( u \subseteq x^ny \) iff \( x = \varepsilon \lor \exists k: u \subseteq x^k y \) as
   \( x = \varepsilon \), then:
   \( u \subseteq x^ny = \top \)
   So assume \( x \neq \varepsilon \).
   If \( u \subseteq x^ny \), then:
   There is some \( k \in \mathbb{N}_0 \) s.t. \( u \leq |x^k y| \).
   Thus \( u^k \subseteq x^ny = x^n x^k y \) iff \( u^k \subseteq x^k y \).
   If \( \exists k: u^k \subseteq x^k y \), then:
   \( u \subseteq x^k y \subseteq x^n x^k y = x^n y \)
3. \( u \rel w^n \sqsubseteq \top \) iff \( u \rel uw \) iff \( \exists w', w'' : w = w'w'' \wedge w'' \sqsubseteq w \wedge u = w''w^k \) as

If \( \exists w', w'' : w = w'w'' \wedge w'' \sqsubseteq w \wedge u = w''w^k \), then:

\[ w = w'' \]

\[ u = w''w^k \]

Thus \( u \rel w''w^k \sqsubseteq w^{k+1} \sqsubseteq w^n \sqsubseteq \top \).

If \( u \rel w^n \sqsubseteq \top \), then:

\[ w = w'' \]

\[ u \neq \varepsilon. \]

Thus there is \( k \in \mathbb{N}_0 \) s.t. \( w^k \sqsubseteq u \sqsubseteq w^{k+1} \) and thus \( u = w''w^k \) for some factorization \( w = w''w^m \) with \( w'' \sqsubseteq w \).

Hence also \( u \sqsubseteq uw = w''w^kw = (w''w')w^k \).

If \( u \sqsubseteq uw \), then:

\[ w = w'' \]

We show by induction on \(|u|\) that \( u \rel w^n \sqsubseteq \top \).

If \(|u| \leq |w|\), then:

\[ w = w'u \]

Thus \( u \rel w \sqsubseteq w^n \sqsubseteq \top \)

So assume \(|u| > |w|\) s.t. \( u = u'w \).

Thus \( u' \sqsubseteq u'w \) as \( u'w = u \rel uw = u'wuw \).

Hence by induction \( u' \rel w^n \sqsubseteq \top \) and thus \( u \rel w^n \sqsubseteq w^n \).

4. \( u \rel w^n \) iff \( u \rel uw \) iff \( \exists v : uw = vu \wedge w^n = v^nu \) as

If \( w = \varepsilon \), then:

\[ u \sqsubseteq w^n = \top \] and \( u \sqsubseteq uw = u \) and \( \exists v : u = uw = vw = v \wedge w^n = v \sqsubseteq u \) are trivially true.

So assume \( w \neq \varepsilon \).

If \( u \rel w^n \sqsubseteq \top \), then:

\[ \exists w', w'' : w = w'w'' \wedge w'' \sqsubseteq w \wedge u = w''w^m \] by preceding result.

Set \( v := w''w \) s.t. \(|v| = |w| > 0\).

Then \( uw = w''w^m = w''w'w^m = vu \).

Hence \( \forall k : w^k \rel v^k u \rel v^nu \sqsubseteq w^nu \sqsubseteq w^n \).

If \( \exists v : uw = vu \), then:

\[ u \rel uw = vu \]

Thus \( u \rel w^n \sqsubseteq \top \).

5. \( u^n v = w^n \) iff \( \exists p, q \in \Sigma^* : pv = vq \wedge u \in p^* \wedge w \in q^* \) as

Wlog, assume \( u \neq \top \) and \( w \neq \) w as otherwise \( u^n v = \top \lor w^n = \top \) s.t. \( u = \top = w \).

Let \( p \) be the primitive root of \( u \), and \( q \) that of \( w \).

Then \( u^n = p^n \) and \( w^n = q^n \) s.t. \( v \rel w^n = q^n \sqsubseteq \top \).

Thus \( \exists q : vq = \hat{q}v \) s.t. \( p^n = q^n \) and \( v = \hat{q}v \).

So \( p^n = q^n \) and thus \( p = \hat{q} \) as both are primitive.

\[ uv = x^n y \]

iff \( u = \varepsilon = x \lor u \neq \varepsilon \neq x \land v \neq \forall k \exists l : u^kv \rel x^k y \sqsubseteq u^kv \).

6. \( u \neq x \neq x \neq u \neq v \neq x \land \forall k \exists l : u^k v \neq x^k y \sqsubseteq \top \) is weakly well-formed.

iff \( u = \varepsilon = x \lor u \neq \varepsilon \neq x \land \forall k \exists l : u^k v \not\sqsubseteq \top \) and \( y \not\sqsubseteq \top \).

iff \( u = \varepsilon = x \lor \exists p, q : u \in q^* \land x \in p^* \land p q \not\sqsubseteq \top \).
Further, let $q$ be the primitive root of $a$. Then $y' \subseteq u^v = q^v \subseteq T$. Hence $\exists p: y'q = py' \wedge q^v \subseteq p^v y' \text{ s.t.:}$

- $py\overline{p}$ $\not\subseteq py \not\subseteq p^v \not\subseteq y'v \not\subseteq y$. 
- $x'\not\subseteq y = u^v = q^v = p^v y = p^v y$ and $x'' = p^v$.

Finally $x \in p^v$ as $x \neq \varepsilon$ and $q$ is primitive (as $y'q = py'$ and $p$ primitive).

If $\forall k \exists \ell: u^k v \subseteq x' y \not\subseteq x^k y' \subseteq u^v$, then:

\[
\forall k \exists \ell: \bigcap_{n \geq 0} y^R(x^k)^n \subseteq y^R(x^k)^n \subseteq v^R(u^k)^n \subseteq v^R(u^k)^n
\]

and thus

\[
\bigcap_{n \geq 0} y^R(x^k)^n \subseteq v^R(u^k)^n
\]

i.e. $u^v \subseteq x' y$ and symmetrically $x'' y \subseteq u^v$ s.t. $x'' y = u^v$.

If $\forall k \exists \ell: u^k v \subseteq x' y \not\subseteq x^k y \subseteq u^v$ directly holds.

If $\exists p, q: u \in q^1 \wedge x \in p^1 \wedge py\overline{p} \not\subseteq y$, then:

Wlog. $|y| \geq |u|$ s.t. $y = y'v$. Then:

$x'' y = p^v y = p^v y'v = q^v y = u^v$.

7. For all $k \geq 1$:

\[\text{lcs}(x, y^k) = \text{lcs}(x, x y^k) = \text{lcs}(x, y^k)\]

as:

Wlog. $|x| \leq |y|$.

If $x = \varepsilon$, then:

\[\text{lcs}(x, y^k) = \text{lcs}(x, y^k) = \text{lcs}(x, y^k)\]

and thus:

\[\text{lcs}(x, y^k) = \text{lcs}(x, y^k) = \text{lcs}(x, y^k)\]

as:

Wlog. $|x| \leq |y|$. 

If $x = \varepsilon$, then:

\[\text{lcs}(x, y^k) = \text{lcs}(x, y^k) = \text{lcs}(x, y^k)\]

So $0 < |x| \leq |y|$.

If $x \not\subseteq xy \land y \not\subseteq xy$, then:

Wlog. $x \not\subseteq xy$.

Then $\text{lcs}(x, y^k) = \text{lcs}(x, y^k) \subseteq x$ for all $k > 0$.

So assume $x \not\subseteq xy \land y \not\subseteq xy$.

Then $\exists \tilde{x}, \tilde{y}: xy = \tilde{y}x \wedge \tilde{y}x = \tilde{x}y$.

If $\tilde{x} = x \not\subseteq \tilde{y} = y$, then:

$x y = y x$ as $x y = \tilde{y}x \wedge \tilde{y}x = \tilde{x}y = x y$.

Hence, $x'' = y''$ as $x, y$ have the same primitive root.

So assume that $\tilde{x} \neq x \wedge \tilde{y} \neq y$.

$y = y'x$ as $|x| \leq |y|$ and $y \not\subseteq xy = \tilde{x}y$.

Hence $\tilde{y} = xy'$ as $\tilde{y}x = xy = x'y x$.

Thus for all $k > 0$:

\[\text{lcs}(x'', y'') = \text{lcs}(x'', y'') = \text{lcs}(\tilde{x}, y x'')\]
Balancedness

9. \( \text{lcs}(u^n, (wu)^n) = \text{lcs}(u^n, w^n)u \) as

\[
\begin{align*}
\text{lcs}(u^n, (wu)^n) &= \begin{cases} 
  u^n & \text{if } uw = wu \land w \neq \varepsilon \\
  w^n & \text{if } uw = wu \land w = \varepsilon \\
  \text{lcs}(uw, wuu) & \text{if } uw \neq wu
\end{cases} \\
= \text{lcs}(u^n, w^n)u
\end{align*}
\]

10. \( \text{lcs}(u^n, w^n, (wu)^n) = \text{lcs}(u^n, w^n) \) as

\[
\begin{align*}
\text{lcs}(u^n, w^n, (wu)^n) &= \text{lcs} \left( \begin{align*}
\text{lcs}(u^n, w^n) \\
\text{lcs}(u^n, (wu)^n) \\
\text{lcs}(u^n, w^n)u
\end{align*} \right) \\
&= \text{lcs} \left( \begin{align*}
\text{lcs}(u^n, w^n) \\
\text{lcs}(u^n, w^n)u
\end{align*} \right) \\
&= \text{lcs}(\text{lcs}(u^n, w^n), w^n) \\
&= \text{lcs}(u^n, w^n)
\end{align*}
\]

The next lemma formalizes that whenever \( L \) is not empty, we can find for any \( x \in L \) a witness \( y \in L \) s.t. \( \text{lcs}(L) = \text{lcs}(x, y) \) which we will use in the following frequently without explicitly referring to this lemma everyday.

► **Lemma 35** (Lemma 2 in the main work). Let \( L \subseteq \Sigma^* \). Then both:
- \( \forall x \in L : \text{lcs}(L) = \text{lcs}(\text{lcs}(x, y) \mid y \in L) \)
- \( \forall x \in L \exists y \in L \forall z \in L : \text{lcs}(L) = \text{lcs}(x, y) = \text{lcs}(x, y, z) \)

**Proof.** Trivially true if \( L = \emptyset \). Therefore, assume \( L \neq \emptyset \). Let \( R = \text{lcs}(L) \) and pick any \( x \in L \). If \( x = R \), we have \( \text{lcs}(x, y) = R \) for all \( y \in L \); so choose any \( y \in L \). Thus, assume \( R \not\subseteq x \).

Let \( L_x = \{ \text{lcs}(x, y) \mid y \in L \} \) and \( S := \text{lcs}(L_x) \). Then \( R \not\subseteq S \) as for all \( y \in L \) we have \( R \not\subseteq y \) and thus \( R \not\subseteq \text{lcs}(x, y) \). But also \( S \not\subseteq R \) as \( \forall y \in L : S \not\subseteq \text{lcs}(x, y) \subseteq y \). Thus, \( R \subseteq L_x \) as \( \forall z \in L_x : R \not\subseteq z \subseteq x \). Then there is some \( y \in L \) s.t. \( R = \text{lcs}(x, y) \); by minimality of \( R \) we trivially have that \( \text{lcs}(x, y, z) = \text{lcs}(x, y) \) for all \( z \in L \).

We recall the definition of the maximal suffix extension:

► **Definition 36** (Definition 22 in the main work). Let \( L \subseteq \Sigma^* \) with \( R = \text{lcs}(L) \).

\[
\text{lcs}(L) := \text{lcs}(z^n \mid zR \in L \setminus \{ R \})
\]

By definition of \( \top \) we have both \( \text{lcs}(\emptyset) = \text{lcs}(\emptyset) = \top \) and \( \text{lcs}(\{ R \}) = \text{lcs}(\varepsilon^n) = \top \).

► **Lemma 37.** Let \( \emptyset \neq L \subseteq \Sigma^* \) and \( R = \text{lcs}(L) \). If \( R \not\in L \), then \( \text{lcs}(L) = \varepsilon \)

**Proof.** As \( R \not\in L \) it exists \( uR, vR \in L \) such that \( R = \text{lcs}(uR, vR) = \text{lcs}(u, v)R \) with \( u \neq \varepsilon \neq v \) and \( \text{lcs}(u, v) = \varepsilon \). Thus, \( \text{lcs}(L) \not\subseteq \text{lcs}(w^n, v^n) = \text{lcs}(u, v) = \varepsilon \).

**A.8 Lemma 24 in the main work**

► **Lemma 38** (Lemma 24 in the main work). Let \( L \subseteq \Sigma^* \) with \( |L| \geq 2 \) and \( R := \text{lcs}(L) \). Fix any \( xR \in L \setminus \{ R \} \). Then there is some \( yR \in L \setminus \{ R \} \) s.t. \( \text{lcs}(L) = \text{lcs}(x^n, y^n) = \text{lcs}(x^n, y^n, z^n) \) for all \( zR \in L \). If \( xy = yx \), then \( R \in L \).

**Proof.**
Note: $x \neq \varepsilon$ as $xR \neq R$.
Let $p$ be the primitive root of $x$.
If $\forall z \in L$: $zx = xz$, then:

$$\forall z \in L: z \in p^*$$

Thus $\text{lces}(L) = p^* = x^* = z^* = \text{lcs}(x^*, z^*)$
$R \in L$ as otherwise $R = \text{lcs}(L) = p' R$ for $p' R$ the shortest word in $L$.

So assume $\exists z \in L: zx \neq xz$, then:

$$z \neq \varepsilon$$

$\text{lces}(L) = \text{lcs}(\text{lces}((x^*), (y^*))) | yR \in L \setminus \{R\}$
$\subseteq \text{lcs}(x^*, z^*) = \text{lcs}(xz, zx)$

Hence, there is some $y \in L \setminus \{R\}$ s.t. $\text{lces}(L) = \text{lcs}(x^*, y^*) = \text{lcs}(xy, yx)$$\Rightarrow$$\Rightarrow$

\section*{A.9 Lemma 26 in the main work}

We split the proof of Lemma 26 into Lemma 39 (union) and Lemma 40 (concatenation).

\section*{Lemma 39 (Lemma 26 (union) in the main work). Let $L, L' \subseteq \Sigma^*$ with $\text{lcssum}(L) = (R, E)$ and $\text{lcssum}(L') = (R', E')$.}

$$\text{lcssum}(L \cup L') = \begin{cases} (R', E') & \text{if } R = \top \\ (R, E) & \text{if } R' = \top \\ (R, \text{lcs}(E, \text{lcs}(E', E\delta)) & \text{if } R' = \delta R' \subseteq \top \\ (R', \text{lcs}(E', \text{lcs}(E, E\delta))) & \text{if } R = \delta R' \subseteq \top \\ (\text{lcs}(R, R'), \varepsilon) & \text{else} \end{cases}$$

\section*{Proof.}

If $R' = \top$:

$$L' = \emptyset$$

$$\text{lcssum}(L \cup L') = \text{lcssum}(L) = (R, E)$$

The case $R = \top$ is symmetric.

Wlog. $R \neq \top \neq R'$ from here on.

If $R \not\subseteq R' \land R' \not\subseteq R$:

$$\text{lcssum}(L \cup L') = (\text{lcs}(R, R'), \varepsilon)$$

Wlog. $R \not\subseteq R' = \delta R$ from here on s.t. $\text{lcs}(L \cup L') = R$.

If $R \not\in L$, then:

$$\text{lces}(L) = \varepsilon$$

and thus $\text{lces}(L \cup L') = \varepsilon = \text{lcs}(\text{lces}(L), \text{lcs}(\text{lces}(L'), \text{lces}(L')\delta))$

So assume $R \in L$.

If $R' \not\in L'$, then:

For suitable $xR', yR' \in L'$.

$$\varepsilon = \text{lces}(L') = \text{lcs}(x^*, y^*) = \text{lcs}(x, y)$$
Thus
\[ \text{lcs}(z R) = \text{lcs}(x, y) = \delta \]
and hence
\[ \text{lcsext}(L \cup L') = \text{lcs}(\text{lcsext}(L), \delta) = \text{lcs}(\text{lcsext}(L), \text{lcsext}(L') \delta) \]

Thus also assume that \( R' \in L' \). Then:
\[ \text{lcsext}(L \cup L') = \text{lcs}(\text{wc}(\delta, \omega), R') \in L, wR \in L) = \text{lcs}(\text{lcsext}(L), \text{lcsext}(L') \delta) \]
As shown before
\[ \text{lcs}(\delta, (z R) \omega) = \text{lcs}(\delta, z \omega) \delta \]

Thus
\[ \text{lcs}(\text{lcs}(\delta, (z R) \omega), (z R) \in L') = \text{lcs}(\delta, \text{lcsext}(L') \delta) \delta \]

Hence again
\[ \text{lcsext}(L \cup L') = \text{lcs}(\text{lcsext}(L), \text{lcsext}(L'), \text{lcsext}(L') \delta) \]

\[ \blacktriangleright \text{Lemma 40} \text{ (Lemma 26 (concatenation) in the main work).} \]
Let \( L, L' \subseteq \Sigma^* \) with \( \text{lcssum}(L) = (R, E) \) and \( \text{lcssum}(L') = (R', E') \).

\[ \text{lcssum}(LL') = \begin{cases} (\top, \top) & RR' = \top \\ (\text{lcs}(R, E') R', \varepsilon) & RR' \not\subseteq \top \land R \not\supseteq E' \\ (RR', E) & RR' \not\subseteq \top \land E' = \top \\ (RR', \text{lcs}(E, \rho(E'R))) & RR' \not\subseteq \top \land R \not\subseteq E' \end{cases} \]

Proof.
If \( R = \top \lor R' = \top \), then:
\[ L = \emptyset \text{ or } L' = \emptyset \text{ s.t. } LL' = \emptyset. \]
\[ \text{lcssum}(LL') = (\top, \top) \]
So assume \( R \not\subseteq \top \neq R' \), i.e. \( L \neq \emptyset \neq L' \).
If \( E' = \varepsilon \), then:
\[ L = \{R'\} \text{ s.t.} \]
\[ \text{lcssum}(LL') = \text{lcssum}(LR') = (RR', E) \]
Thus assume also that \( E' \neq \top \text{ s.t. } |L'| \geq 2. \)
Fix \( xR', yR' \in L' \setminus \{R'\} \text{ s.t.} \)
\[ E' = \text{lcsext}(z R') \in L' \}
Consider
\[ \text{lcs}(LL') = \text{lcs}(wRzR' | wR \in L, zR' \in L') \]
If \( E' = \varepsilon \), then:
\[
\text{lcs}(LL') = \text{lcs}(wRzR' | wR \in L, zR' \in L') \subseteq \text{lcs}(wRzR', wRyR' | wR \in L) = \text{lcs}(xR', yR') = R'
\]

and

\[
\text{lcssum}(LL') = \text{lcs}((wRz)'' | wR \in L, zR' \in L') \subseteq \text{lcs}(wRx)'', (wRy)'' | wR \in L) \subseteq \text{lcs}(x, y) = \varepsilon
\]

and hence

\[
\text{lcsext}(LL') = \left(\text{lcssum}(LL') \right) = (R', \varepsilon) = (\text{lcssum}(R, E') R', \varepsilon)
\]

So assume \(E' \neq \varepsilon\) s.t. \(R' \in L'\). Then

\[
\text{lcs}(LL') = \text{lcs}(wRzR' | wR \in L, zR' \in L') = \text{lcs}\left(\text{lcs}(wR, wRz) | wR \in L, zR' \in L'\right) R'
\]

Consider then:

\[
\text{lcs}(R, E') = \text{lcs}(R, \text{lcsext}(L')) = \text{lcs}(R, \text{lcs}(z'' | zR' \in L')) = \text{lcs}(R, Rz, Rz)
\]

If \(R \not\subseteq E'\), then:

\[
R \not\subseteq \text{lcs}(R, E') = \text{lcs}(R, Rx, Rz)
\]

Thus also \(\text{lcs}(LL') = \text{lcs}(RL') = \text{lcs}(R, E') R' \not\subseteq RR'\).

So \(\text{lcsum}(LL') = (\text{lcs}(R, E') R', \varepsilon)\).

So assume \(R \not\subseteq E'\), i.e. \(\forall zR' \in L' \backslash \{R'\} : R \not\subseteq z''\) s.t.:

\[
\forall zR' \in L' \backslash \{R'\} \exists z : Rz = zR \land z'' = \hat{z} \land \hat{z} R
\]

Hence:

\[
LL' = \{wRz' R | wR \in L, zR' \in L'\} = \{wRzRR' | wR \in L, zR' \in L', Rz = zR\}
\]

and \(\text{lcs}(LL') = RR'\).

Consider then

\[
\text{lcsext}(LL') = \text{lcs}(\rho(wRzR' RR')'' | wR \in L, zR' \in L', wz \neq \varepsilon) = \text{lcs}\left(\text{lcs}(w'' | wR \in L \backslash \{R\})\right) = \text{lcs}\left(\text{lcs}(w'' | wR \in L, zR' \in L' \backslash \{R'\}, Rz = \hat{z} R) \right) = \text{lcssext}(L) \right) \]

If \(E = \top\), i.e. \(L = \{R\}\), then:

\[
\text{lcsext}(LL') = \text{lcs}(\hat{z}'' | zR' \in L', z \neq \varepsilon, Rz = \hat{z} R) \subseteq \text{lcs}(\hat{z}'' | zR' \in L', z \neq \varepsilon) = \text{lcsext}(L)
\]

\[
\text{lcsext}(LL') = \text{lcs}(E, \rho(E' RR')) = \text{lcs}(E, \rho(E' RR'))
\]

So \(\text{lcsum}(LL') = (RR', \text{lcs}(E, \rho(E' RR'))\)
Balancedness

So assume that also $|L| \geq 2$.

Fix $uR, vR \in L \setminus \{R\}$ s.t.

$$\forall wR \in L: E = \text{lces}(L) = \text{lcs}(u^\omega, v^\omega) = \text{lcs}(u^\omega, v^\omega, w^\omega)$$

If $E = \varepsilon$, then:

$$\text{lces}(LL') = \varepsilon = \text{lcs}(E, \rho(E' \overline{R}))$$
$$\text{lcssum}(LL') = (RR', \text{lcs}(E, \rho(E' \overline{R})))$$

Thus also assume $\text{lces}(L) \neq \varepsilon$ and thus also $R \in L$ s.t.:

$$\text{lces}(LL') = \text{lcs}(\text{lces}(LL') | wR \in L, zR' \in L', wz \neq \varepsilon)$$

$$\text{lces}(LL') = \text{lcs}(\text{lces}(\text{llcsext}(L), zR' \in L' \setminus \{R'\}, zR = \varepsilon R))$$

As shown

$$\text{lcs}((w\hat{z})^\omega, \hat{z}^\omega, w^\omega) = \text{lcs}(\varepsilon, \hat{z}^\omega) \cong \text{lcs}(E, \rho(E' \overline{R})))$$

Thus

$$\text{lces}(LL') = \text{lcs}(E, \rho(E' \overline{R}))$$

and again $\text{lcssum}(LL') = (RR', \text{lcs}(E, \rho(E' \overline{R})))$.

\[\triangleright\]

**Corollary 41.** $\approx_{\text{lcs}}$ is a congruence relation on the language semiring $\langle 2^\Sigma^*, \cup, \cdot \rangle$ s.t. the quotient w.r.t. $\approx_{\text{lcs}}$ is a semiring again with the projection $\text{lcssum}$ a homomorphism.

Every $L \subseteq \Sigma^*$ is thus $\approx_{\text{lcs}}$-equivalent to some sublanguage $T_{\text{lcs}}(L) \subseteq L$ consisting of at most three words:

**Fact 1.** For $L \subseteq \Sigma^*$ we define the following sublanguage $T_{\text{lcs}}(L) \subseteq L$

$$T_{\text{lcs}}(L) := \begin{cases} L & \text{if } |L| \leq 2 \\ \{R, xR, yR\} & \text{if } \{R, xR, yR\} \subseteq L \land \text{lces}(L) = \text{lcs}(x^\omega, y^\omega) \quad \text{with } R := \text{lcs}(L) \\ \{xR, yR\} & \text{if } R = \text{lcs}(xR, yR) \land R \notin L \land \{xR, yR\} \subseteq L \end{cases}$$

Then independent of the concrete choice of $x$ and $y$ (up to the given side constraints):

$$L \approx_{\text{lcs}} T_{\text{lcs}}(L)$$

**A.10 Lemma 4 in the main work**

**Lemma 42** (Lemma 4 in the main work). A context-free grammar $G$ is $w$ if and only if $G$ is $b$ with $r_S = \varepsilon$ for $S$ the axiom of $G$.

**Proof.**
Let $S$ be the axiom of $G$.

Wlog, $G$ is reduced to the nonterminals which are reachable from $S$ and which are productive.

Assume first that $G$ is well-formed. Then:

For every nonterminal $X$ of $G$ we can define its left-context $L_X = \{ \alpha \in B^* \mid S \rightarrow_G^* \alpha X \beta \}$.

As $L := L(G)$ is well-formed and every $\alpha \in L_X$ is a prefix of some word of $L$, also $L_X$ is well-formed; hence $m_X := \min(\Delta(L_X)) \geq 0$ is defined. Fix any $\lambda_X \in L_X$ with $\Delta(\lambda_X) = m_X$.

Then for any $\gamma = \gamma' \gamma'' \in L_X$ also $\lambda_X \gamma'$ is a prefix of a word of $L$, thus well-formed, and therefore $\Delta(\lambda_X \gamma') \geq 0$. It follows that $d_X = \min\{-\Delta(\gamma') \mid \gamma'' \in L_X\} \leq m_X$.

In particular, there is a $\gamma_X = \gamma' \gamma'' \in L_X$ s.t. $-\Delta(\gamma') = d_X$; as $L$ is well-formed, $L_X$ has to be weakly well-formed s.t. $\gamma_X \triangleq \frac{r_X}{s_X}$ and $s_X = \frac{r_X}{x}$. Hence, for every $\gamma \in L_X$ we have $\gamma \triangleq s_X \gamma$ and $\lambda_X \gamma \triangleq \frac{x}{X} s_X$ well-formed, i.e. $y \subseteq r_X$ and thus $x \in L_X$ is well-formed.

In particular, we have $\tau_S = \varepsilon$ for the axiom $S$.

Assume that $G$ is bwf and thus by definition also nonnegative. Then:

We fix for every $X$ any $r_X \in A^*$ s.t. $r_X L_X$ is well-formed; hence, $L_X$ is weakly well-formed and $d_X = \max\{|y| \mid \gamma \in L_X, \rho(\gamma) = \frac{x}{x}\}$.

Then $\tau_X := \max\{\frac{\varepsilon}{x}, |y| \mid \gamma \in L_X, \rho(\gamma) = \frac{x}{x}\} \subseteq r_X$ is well defined with $d_X = |r_X|$.

As $G$ is also nonnegative, we have $d_S = 0$ resp. $r_S = \varepsilon$ and thus $L = L_S$ is well-formed.

---

A.11 Lemma 5 in the main work

- **Fact 2.** Let $G$ be a context-free grammar over the nonterminals $X$. Define $G_p$ by the following rules:
  - If $X \rightarrow_G Y Z$, then $X \rightarrow_{G_p} Y P, X \rightarrow_{G_p} P Y Z$ and $X \rightarrow_{G_p} Y Z$.
  - If $X \rightarrow_G Y$, then $X \rightarrow_{G_p} Y, X \rightarrow_{G_p} P Y$ and $X \rightarrow_{G_p} Y$.
  - If $X \rightarrow_G \pi V$, then $X \rightarrow_{G_p} \pi V$ and $X \rightarrow_{G_p} \pi V$

Then $L_X(G) = L_X(G_p)$ and $L_{X_p}(G_p) \cup \{\varepsilon\} = \operatorname{Prf}(L_X(G))$. In particular, we can construct $G_p$ in time polynomial in the size of $G$.

- **Lemma 43.** Let $L = L(G) = \operatorname{Prf}(L(G))$ be a prefix-closed context-free language. We can decide in time polynomial in $G$ whether there is a word $\alpha \in L$ s.t. $\Delta(\alpha) < 0$.

  **Proof.**

  Let $N$ be the number of nonterminals of $G$. Assume there is a word $\alpha \in L$ with $\Delta(\alpha) < 0$, then wlog. $\Delta(\alpha) = -1$ as $L$ is prefix-closed. Pick any shortest such $\alpha \in L$ with $\Delta(\alpha) = -1$.

  If $\alpha$ has a derivation tree of height at most $N$, then we simply apply standard fixed-point/Kleene iteration to the operator $F$ obtained from the rewrite rules of $G$ via the homomorphism $\Delta$ over the tropical semiring

  
  $$F(X) := \min(\Delta(Y) + \Delta(Z), \Delta(Y), \Delta(\gamma) \mid X \rightarrow_G Y Z, X \rightarrow_G Y, X \rightarrow_G \gamma)$$

  Then $F^N(\alpha) = \min\{\Delta(\beta) \mid \beta \in L^{\leq N}\} \leq \Delta(\alpha) = -1$ with $L^{\leq N}$ all words of $L$ that possess a derivation tree of height at most $N$.

  Assume thus that every such $\alpha$ has a derivation tree of height at least $N + 1$. Pick a longest path from the root to a leaf in such a derivation tree, and moving bottom-up along this path, pick the first nonterminal $X$ occuring a second time in order to obtain a factorization $\alpha = \beta \rho \gamma \delta$ s.t. $\beta \rho^k \gamma \delta \delta \in L$ for all $k \in \mathbb{N}_0$ and $\rho \gamma \neq \varepsilon$; in particular, note that $\rho \gamma \in L^{< N}_X$. 

Then $-1 = \Delta(\alpha) = \Delta(\beta\gamma\delta) + \Delta(\rho\gamma\delta)$ and $\Delta(\beta\gamma\delta) \geq 0$; otherwise there would be a prefix $\pi$ of $\beta\gamma\delta$ with $\Delta(\pi) = -1$ contradicting the minimality of $\alpha$. Hence, $\Delta(\rho\gamma\delta) \leq -1$.

Thus, we only need to decide whether there is a pumpable derivation tree $X \rightarrow^{\leq N}_G \rho X \varrho$ of height at most $N$ s.t. $\Delta(\rho\varrho) < 0$. This can be done by transforming the rewrite rules $X \rightarrow Y Z$ into weighted edges $X \xrightarrow{F^N(\infty)} Z$ and $X \xrightarrow{F^N(\infty)} Y$, and then check for negative cycles in this graph. (This amounts to take the derivative of $F$ at $F^N(\infty)$.)

Lemma 44 (Lemma 5 in the main work). Let $L = L(G)$ be wff. Let $X$ be some nonterminal of $G$. Derive from $G$ the CFG $G^\rho$ s.t. $\text{Prf}(L_X) = L(G^\rho_X)$. Let $r_X \in A^*$ be the shortest word s.t. $r_X L_X$ is wff. Then:

1. $r_X$ is also the shortest word s.t. $r_X \text{Prf}(L_X)$ is wff.
2. There is a shortest $\alpha \in \text{Prf}(L_X)$ s.t. $\rho(\alpha) = \overline{r_X}$.
3. Every shortest $\alpha \in \text{Prf}(L_X)$ with $\rho(\alpha) = \overline{r_X}$ has a derivation tree w.r.t. $G^\rho_X$ that does not contain any pumping tree and thus has height bounded by the number of nonterminals of $G^\rho_X$.
4. A SLP for $r_X$ can be computed in PTIME.

Proof.

1. As $L_X \subseteq \text{Prf}(L_X)$, we only need to show that $u_X \text{Prf}(L_X)$ is still wff.

For every $\alpha' \in \text{Prf}(L_X)$ there is some $\alpha \in L_X$ s.t. $\alpha = \alpha'\alpha''$ (by definition). As $L$ is wff, $L_X$ is wff, hence $\alpha$ is wff, and thus $\alpha'$ and $\alpha''$ are wff, too. Hence, $\rho(\alpha') = \overline{s}$, $\rho(\alpha'') = \overline{v}$, and $\rho(\alpha) = \overline{y}$ for some $s, u, v, x, y \in A^*$ s.t. $s \overline{p} v \leq p y$. If $s = s', u$; then $\overline{s} = \overline{s}$; else $u = u'$ and $\overline{s} = \overline{s} \overline{u}$.

2. There is some $\beta \in L_X$ s.t. $\rho(\beta) = \overline{x} x y$ for some $y \in A^*$. Then there is some prefix $\beta' \subseteq \beta$ s.t. $\rho(\beta') = \overline{x} x y$. By definition, $\beta' \in \text{Prf}(L_X)$. Hence, there is also some shortest $\alpha \in \text{Prf}(L_X)$ s.t. $\rho(\alpha) = \overline{x} x y$.

3. Let $\alpha \in \text{Prf}(L_X) = L(G^\rho_X)$ be a shortest word s.t. $\rho(\alpha) = \overline{x} x y$. Assume that there is some factorization $\alpha = \beta \gamma_1 \rho \delta$ s.t. $\beta \gamma_2 \gamma_3 \delta \in L(G^\rho_X)$ for all $k \in N_0$.

Consider $k = 0$ and let $\rho(\beta \gamma \delta) = \overline{x} s$. If $r = u_X$, then there would be a prefix of $\beta \gamma \delta$ that would reduce to $\overline{x} x y$ contradicting our assumption that $\alpha$ is a shortest such word. Hence, $\overline{x} \overline{s} \overline{x} \overline{y}$. Thus $\Delta(\beta \gamma \delta) \geq -|r| > -|u_x|$. Note that $-|u_x| = \Delta(\alpha) = \Delta(\beta \gamma \delta) + \Delta(\rho \varrho)$. Thus, $\Delta(\rho \varrho) < 0$; a contradiction to the wffness of $u_X \text{Prf}(L_X)$.

4. Split every nonterminal $Y$ of $G^\rho_X$ into $N + 1$ copies $Y_0, \ldots, Y_N$, split every rule $Y \rightarrow UV$ into the rules $Y_{i+1} \rightarrow U_i V_i | Y_i$, and derive from every rule $Y \gamma$ the rule $Y_0 \gamma$. In other words, unfold $G^\rho_X$ into an acyclic grammar that generates exactly all derivation trees of height at most $N$.

We compute inductively for every nonterminal a pair of SLPs representing a wff word $\overline{x} v$ as follows:

For every rule $Y_0 \rightarrow \gamma \in B$, we choose either $u = \gamma$, $v = \varepsilon$ or $u = \varepsilon$, $v = \gamma$ such that $\overline{v} v = \gamma$.

For every rule $Y_{i+1} \rightarrow U_i V_i$, we have inductively computed SLPs for $U_i$ and $V_i$ representing words $\overline{u} s$ and $\overline{u} v$, respectively. Then we can compute SLPs representing the reduce $\rho(\overline{s} \overline{u} \overline{v})$; either $s = s' u$ (i.e. $|u| \geq |u|$) or $u = u'$ (i.e. $|u| \leq |u|$), i.e. we simply have to restrict and then concatenate the respective SLPs. For the rule $Y_{i+1} \rightarrow Y_i$ there is nothing to do.

We are thus left for $Y_{i+1}$ with a family of SLPs representing words $\overline{w} v_i$: w.l.o.g. assume $|u_i| \geq |u|$ for all $i$; as for every derivation $X_N \rightarrow^* \alpha Y_{i+1} \gamma$ we need to have that $\alpha \overline{w} v_i \gamma$ is wff for every $i$, we also have that $\alpha \overline{w} u_0 \overline{v}_i \gamma$ is wff for every $i$. We thus may normalize all SLP pairs by means of $\overline{w} v_i \rightarrow \overline{w} \rho(u_0 \overline{w}) v_i$. As we want to maximize the descent, we then assign to $Y_{i+1}$ the pair of SLPs encoding $\overline{w}$ and the shortest of all $\rho(u_0 \overline{w}) v_i$. This amounts to a constant amount of SLP operations per rule of the unfolded grammar.
A.12 Reduced LCS of simple linear wf languages

The following Lemmas 50 to 52 state the central combinatorial results underlying the proof of Lemma 30. They are concerned with the reduced lcs of simple linear grammars of the form

\[ S \rightarrow \alpha X \beta \quad X \rightarrow \sigma_1 X \tau_1 \mid \ldots \mid \sigma_k X \tau_k \mid \gamma \quad (\alpha, \beta, \sigma_i, \tau_i, \gamma \in B^*) \]

which arise from the factorization of derivation trees:

Given (i) a derivation tree of a context-free grammar \( G \) that yields the word \( \kappa \), (ii) a path within this tree, and (iii) a specific nonterminal \( X \) of \( G \), we may factorize \( X \) into the product of \( (\text{word}) \) contexts (finite words with a “hole” which represent a pumping tree w.r.t. \( G \)) \( (\alpha, \beta) \), \((\sigma_1, \tau_1)\), \ldots, \((\sigma_k, \tau_k)\) and a single word \( \gamma \) s.t. \( S \rightarrow_G \alpha X \beta \), \( X \rightarrow_G \sigma_i X \tau_i \), and \( X \rightarrow_G \gamma \). We denote such factorizations by simply writing \( \kappa = (\alpha, \beta)(\sigma_1, \tau_1) \ldots (\sigma_k, \tau_k)\gamma \). Concatenation of contexts with contexts resp. words is thus defined by means of substituting the right operand into the “hole” of the context, i.e. \((\sigma, \tau)(\mu, \nu) = (\sigma \mu, \nu \tau) \) and \((\sigma, \tau)\gamma = \sigma \tau \gamma\).

Such a factorization then induces the \textit{simple linear language}

\[(\alpha, \beta)[(\sigma_1, \tau_1) + \ldots + (\sigma_k, \tau_k)]^* \gamma := \{\alpha \sigma_{i_1} \ldots \sigma_{i_t} \tau_{i_t} \ldots \tau_{i_1} \gamma \mid i_1 \ldots i_t \in \{1, \ldots, k\}^*\}\]

which is generated by the \textit{simple linear grammar}

\[ S \rightarrow \alpha X \beta \quad X \rightarrow \sigma_1 X \tau_1 \mid \ldots \mid \sigma_k X \tau_k \mid \gamma \]

and is thus always a sublanguage of \( L(G) \). Assuming that \( G \) is well-formed, we show in the proof of Lemma 30 that we can rewrite each rule so that the simple linear grammar takes the form

\[ S \rightarrow uX \quad X \rightarrow s_1 X \tau_1 \mid \ldots \mid s_k X \tau_k \mid w \quad (u, v, w, s_i, r_i, t_i \in A^*, \tau_1 = \tau_2 t_{r_1} \lor \tau_1 = \tau_2 t_{r_1}) \]

where both grammars generated the same language after reduction, and there is one-to-one correspondence of the rewrite rules s.t. the derivations of both grammars are in bijection. For the proof of Lemma 30, it suffices to consider where \( k = 2 \), i.e. derivation tree has been factorized into two pumping trees.

The central observation in Lemmas 51 and 52 is that, if at least one of the contexts \((s_i, \tau_i)\) is negative, i.e. \( \tau_i \not\parallel t_{r_i} \) with \( t \neq \varepsilon \), then the simple linear well-formed \( L \) can be normalized to a regular language \( A \) whose lcs and lcsext are already determined by \((u, v)\varepsilon\) and \((u, v)(s_i, \tau_i)\varepsilon\). See also Example 45.

\begin{itemize}
  \item \textbf{Example 45.} Consider the linear language \( L' \) given by the rules \( S \rightarrow uX \) and \( X \rightarrow sX \tau tr \mid \varepsilon \) where we assume that the language is wf with \( t \neq \varepsilon \) and, for the sake of this example, also \( |tr| > |s| \). As \( u^k s^k \tau^k t^k r^k \) is wf for all \( k \in \mathbb{N} \), we have \( (s^\omega)^R = (t^\omega r^\omega)^R \) i.e. there is conjugate \( p \) of the primitive root \( q \) of \( t \) s.t. (i) \( qr = rp \), (ii) \( s = p^n \), (iii) \( t = q^n \), and (iv) \( m \geq n \) for suitable \( m, n \in \mathbb{N}_0 \). Property (iv) has to hold as otherwise we could generate a negative word. Further as \( |tr| > |s| \) we have \( trs \not\parallel r\bar{p}^m s \) s.t. \( r = r'^p m^{-n} \), \( qr' = r'^p p \), and \( u = u' r' \) as \( us\bar{t} t r \) is wf. We thus may replace \( X \rightarrow sX \tau tr \) with \( X \rightarrow p^m - n X \) as

\[
\begin{align*}
us^{k+1} t^{k+1} r^{k+1} & \not\parallel u' r'(p^m)^{k+1} p^m - n p^m - n (q^n)^{k+1} r' p^m - n \\
& \not\parallel u' r'(p^m)^{k+1} p^m - n p^m - n (q^n)^{k+1} r' p^m - n \\
& \not\parallel u'(q^n)^{k+1} x^{m-n} r' \not\parallel u'(q^m - n)^{k+1} r' \not\parallel u(p^m - n)^{k+1}
\end{align*}
\]
\end{itemize}
Balancedness

s.t. we obtain a regular language $L' \subseteq A^*$ whose derivations are in bijection with those of $L$. Now, $\text{lcssum}(L)$ is already determined by $u$ and $u_p$ which in turn implies that $u$ and $usu^{tr}$ determine $\text{lcssum}(L)$. In case of multiple contexts $(s_j, \tau_j)$ the existence of one context of the form $(s_i, \overline{r_i} t_i r_i)$ enforces that all contexts have to be compatible with the primitive root of $t_i$ which subsequently allows us to replace every rule $X \rightarrow s_i \tau_i X$ by a $\bar{\rho}$-equivalent rule $X \rightarrow p^k X$ over $A$.

On the other hand, if both contexts $(s_i, \tau_i)$ are nonnegative, i.e. $\tau_i = \overline{r_i} t_i r_i$ for $i = 1, 2$, then Lemma 43 shows that the $\text{lcs}$ and $\text{lcsExt}$ of the simple linear well-formed language $L$ is already determined by $(u(v), \varepsilon)$ and either some word $(u(v), (s_i, \tau_i) \varepsilon)$ or some word $(u(v), (s_i, \tau_i) \varepsilon)$ for some $i \in \{1, 2\}$ with the important point that $j$ can be chosen arbitrarily from $\{1, 2\}$ — this is central to the proof of Lemma 43. See also Example 46.

**Example 46.** Consider the well-formed language $L = (uR, \varepsilon) [(s_1, \tau_1 t_1 r_1) + (s_2, t_2)]^\varepsilon (u, s_1, s_2, r, t_1, t_2, R \in A^*)$ where we assume that (i) $R = t_1 r_1 = \text{lcs}^\theta(L)$, (ii) $r = r' s_1$ with $r' \neq \varepsilon$, (iii) $r' \not\subseteq r s_2$ (i.e. there is no conjugate of $s_2$ w.r.t. $r$), and (iv) $t_2 = t_2'$ $w$. If $u R s_1 t_1 r t_2 = u t_1 s_1 r' s_2' t_1 r t_2$ is w.f., there is some conjugate $\hat{s}_1$ s.t. $s_1' = \hat{s}_1 r'$. Subsequently, there still has to exist a conjugate $\hat{s}_2$ of $s_2$ with $r' \not\subseteq \hat{s}_2 r'$ as $u R s_2 t_1 r t_2 \not\subseteq u t_1 \hat{s}_1 r' s_2' t_1 r t_2$ is w.f. These conjugates allow us to remove the closing brackets, but only by splitting the simple language depending on which contexts are used in a derivation:

$$
(uR, \varepsilon) [(s_1, \tau_1 R) + (s_2, t_2)]^\varepsilon (u, s_1, s_2, r, t_1, t_2, R \in A^*)
$$

1. If $\rho \not\subseteq r s_2$, then $s_2$ is a witness w.r.t. $(u, \varepsilon)$, and $u R s_1 t_1 r t_2 = u t_1 \hat{s}_1 r' s_2' t_1 r t_2$ is w.f.

2. If $s_1 \not\subseteq \hat{s}_2 r'$, then $(uR, \varepsilon) [(s_1, \tau_1 r t_1) = u t_1 s_1 r' s_2' t_1 r t_2]$ is w.f.

Before proving Lemmas 50 to 52 we need the additional Lemmas 47 and 48 for the case without closing brackets (i.e. $r_i = \varepsilon$). Both lemmas are stronger versions of the analogous results for the $lcp$ as presented in [9]. Most importantly, both lemmas now state that, if e.g. $(u, \varepsilon) (s_1, t_1) = w$, then also $(u, \varepsilon) (s_1, t_1) (s_2, t_2) = w u s_1 s_2 t_1 t_2$ is a witness w.r.t. $u w i e$; i.e. only the outer context resp. pumping tree matters in the end.

**Lemma 47.** Let $L = (u, \varepsilon) [(s_1, t_1) + (s_2, t_2)]^\varepsilon$. Then $\text{lcs}(L) = \text{lcs}(u u s_1 t_1, u s_1 s_2 t_2)$.

If $s_1 = \varepsilon$, then $u s_1 s_2 t_1 t_2$ is not required.

**Proof.** Let $R := \text{lcs}(L)$ and $u = u R$. It exists $\hat{t}_i$ such that

$$
R t_i = \hat{t}_i R
$$

If $t_i = \varepsilon$, then set $\hat{t}_i = \varepsilon$; otherwise we have $R \subseteq \hat{t}_i$ as $R \subseteq u s_1 t_1 k$ for all $k \geq 0$.

**Claim.** If $R \subseteq t_1 = t_1' R \land R \subseteq t_2 = t_2' R$ then $\text{lcs}(L) = \text{lcs}(u u s_1 t_1, u s_2 t_2)$.

\[
\begin{pmatrix}
  u & u s_1 t_1 \\
  u s_1 s_2 t_2 & u s_2 t_2
\end{pmatrix}
\]

$R$. 

Thus, assume w.l.o.g. that \( t_1 \preceq R = \hat{R}t_1 \) from here on. Then,
\[
R = \hat{R}t_1 = \dot{i}_1 \hat{R}
\]
If \( t_1 = \varepsilon \), we set \( \dot{i}_1 = \varepsilon \) and \( R = \hat{R} \); otherwise \( \hat{R}t_1 t_1 = Rt_1 = \dot{i}_1 R = \dot{i}_1 \hat{R}t_1 \) (cancel \( t_1 \) from the left). Additionally, there exists \( \dot{s}_1 \) such that
\[
\hat{R}s_1 = \ddot{s}_1 \hat{R}
\]
If \( s_1 = \varepsilon \), we set \( \dot{s}_1 = \varepsilon \); otherwise \( \hat{R} \sqsubseteq \hat{R}s_1 \) as \( \hat{R}t_1 = R \sqsubseteq u \hat{s}_1 t_1 = u' \hat{R}t_1 s_1 t_1 = u' \dot{i}_1 \hat{R}s_1 t_1 \).

\[\mathcal{Q}\text{ Claim. } \text{If } R \sqsubseteq t_2 = t_2^* R = t_2^* \dot{i}_1 \hat{R} (t_2^* \neq \varepsilon) \text{ then the lemma follows.} \]

Proof. We have
\[
\begin{align*}
(u, \varepsilon)(s_1, t_1)^* \varepsilon &= (u' \ddot{i}_1 \hat{R}, \varepsilon)(s_1, t_1)^* \varepsilon \\
&= (u' \ddot{i}_1 s_1, R)(\ddot{s}_1, \ddot{i}_1)^* \varepsilon \\
(u, \varepsilon)(s_1, t_1)^* (s_2, t_2)[(s_1, t_1) + (s_2, t_2)]^* \varepsilon &= (u, \varepsilon)(s_1, t_1)^* (s_2, t_2)^* [[(s_1, t_1) + (s_2, t_2)]^* \varepsilon \\
&= (u, R)(s_1, \ddot{i}_1)^* (s_2, t_2)^* \varepsilon \\
\end{align*}
\]
Therefore \( \varepsilon \overset{1}{=} \mathcal{lc}(u', \ddot{i}_1 s_1, \dot{s}_1 \ddot{i}_1^*, t_2^*, t_2^*) \). If \( \mathcal{lc}(u', \ddot{i}_1 \dot{s}_1) = \varepsilon \) then \( u s_1 t_1 \) is a witness. If \( \mathcal{lc}(u', \ddot{i}_1 \dot{s}_1) = \varepsilon \) and \( \ddot{i}_1 \neq \varepsilon \) then \( \mathcal{lc}(u', \dot{s}_1 \ddot{i}_1') = \varepsilon \) and \( u \dot{s}_1 \ddot{i}_1 t_1 \) is a witness and \( u s_2 \dot{s}_1 t_1 t_2 \).

Note that if \( \ddot{i}_1 = \varepsilon \) then \( \dot{s}_1 \neq \varepsilon \) and we are in the first case where \( u \dot{s}_1 t_1 \) is a witness. If \( \mathcal{lc}(u', t_2') = \varepsilon \) then \( u \dot{s}_2 t_2 \) is a witness.

We therefore consider the case that \( t_2 \overset{a}{=} R = \hat{R}t_2 \) from here on. Then
\[
R = \hat{R}t_2 = \dot{i}_2 \hat{R} \text{ and } u = u' R = u' \hat{R}t_2 = u' \dot{i}_2 \hat{R}
\]
as \( \hat{R} \dot{t}_2 = R \dot{t}_2 = \dot{i}_2 R = \dot{i}_2 \hat{R} \dot{t}_2 \) (cancel \( t_2 \) from the left). W.l.o.g. we assume that \( |t_1| \leq |t_2| \).

\[\mathcal{Q}\text{ Claim. } \text{We find the following conjugates} \]
1. \( \exists \ddot{z}_2: \hat{R}s_2 = \ddot{s}_2 \hat{R} \)
2. \( \exists z: t_2 = z t_1 \wedge R = \hat{R}z \)
3. \( \exists \ddot{z}: \hat{R}z = \ddot{z} \hat{R} \)
4. \( \exists \ddot{s}_1: \hat{R}s_1 = \ddot{s}_1 \hat{R} \)
5. \( \exists \ddot{i}_1: \hat{R}t_1 = \ddot{i}_1 \hat{R} \)
6. \( \ddot{i}_2 = \ddot{z} \ddot{i}_1 = \ddot{i}_1 \ddot{z} \)

Proof. 1. We have \( \hat{R}t_2 = R \overset{a}{=} u \dot{s}_2 t_2 = u' \hat{R}t_2 s_2 t_2 = u' \dot{i}_2 \hat{R} \dot{s}_2 t_2 \) and therefore \( R \overset{a}{=} \hat{R} \dot{s}_2 \).
2. We have \( R = \hat{R}t_2 = \hat{R} \dot{t}_2 \) and \( |t_1| \leq |t_2| \) and therefore \( \hat{R}z t_1 = \hat{R}z \).
3. We have \( \ddot{i}_2 \hat{R} = \ddot{R}t_2 = \hat{R}t_2 = \dot{i}_1 \hat{R} = \dot{i}_1 \hat{R}z \) and therefore \( R \overset{a}{=} \ddot{R}z \).
4. If \( t_2 = \varepsilon \), then \( \hat{R} = R = \ddot{R} = \hat{R} \) and \( \ddot{s}_1 = \ddot{s}_1 = \ddot{s} \). Otherwise, we have \( \hat{R}_1 \overset{a}{=} \ddot{R}t_1 = R \dot{z} t_1 = \ddot{R}t_2 = R \overset{a}{=} u \dot{s}_1 t_1 = u' \dot{i}_2 \hat{R} \dot{s}_1 t_1 \) and therefore \( R \overset{a}{=} \hat{R} \dot{s}_1 \).
5. We have \( \ddot{i}_2 \hat{R} = \ddot{R}t_2 = R \overset{a}{=} \dot{i}_1 R = R \overset{a}{=} \hat{R}t_2 t_1 = \dot{i}_2 \hat{R} \hat{R}t_1 \) and therefore \( R \overset{a}{=} \hat{R} \hat{R}t_1 \).
6. We have \( \hat{t}_2 \hat{R} = R = \hat{R} t_1 = \hat{t}_1 \hat{R} = \hat{t}_1 \hat{R} \) and thus \( \hat{t}_2 = \hat{t}_1 \). Additionally, \( t_2 \hat{R} = R = \hat{R} t_1 = \hat{R} t_2 = \hat{R} t_1 = \hat{R} t_1 \hat{R} \) holds and thus \( t_2 = \hat{t}_1 \).

Using these conjugates we obtain:
\[
(u, \varepsilon)(s_1, t_1)^+ \varepsilon = (u' t_1 \hat{R}, \varepsilon)(s_1, t_1)^+ \varepsilon \\
= (u' t_1 s_1, R)(s_1, t_1)^+ \varepsilon \\
= u' t_1 s_1 R \\
+ (u' t_1 \hat{s}_1 \hat{t}_1 R)(s_1, t_1)^+ \varepsilon \\
= (u, \varepsilon)(s_1, t_1)^+ (s_2, t_2)[(s_1, t_1) + (s_2, t_2)]^+ \varepsilon
\]

"extracting" the \( \text{lcs} R \) by substituting the corresponding conjugates of \( u, s_1, t_1 \).

If \( u s_1 t_1 \) or \( u s_2 t_2 \) is a witness then the claim of the lemma follows. Thus, assume that neither \( u s_1 t_1 \) nor \( u s_2 t_2 \) is a witness w.r.t. \( u, i.e.

\[
\text{lcs}(u', \hat{t}_1 s_1, \hat{z}_1 \hat{s}_2) \neq \varepsilon
\]

Wlog. \( t_2 \neq \varepsilon \) and thus also \( \hat{t}_2 \neq \varepsilon \) as otherwise \( t_1 = \varepsilon \) as \( 0 = |t_2| \geq |t_1| \) s.t. \( R = \hat{R} = \hat{R} \) and \( \hat{s}_1 = \hat{s}_1 \). Then \( L = u(s_1 + s_2)^+ = u'(s_1 + \hat{s}_2)^+ R \) and thus \( \text{lcs}(u', \hat{s}_1, \hat{s}_2) = \varepsilon \). Therefore \( u s_1 t_1 = u s_1 \) or \( u s_2 t_2 = u s_2 \) would be a witness.

\( \triangleright \) Claim. If \( t_1 = \varepsilon \) then the lemma follows.

Proof. We have \( \hat{t}_1 = \hat{t}_1 = \varepsilon \) and \( \hat{R} = R \) and \( \hat{s}_1 \neq \varepsilon \neq \hat{s}_1 \) and \( \hat{t}_2 = \hat{z}_1 \hat{z} = \hat{z} \neq \varepsilon \) s.t.
\[
(u, \varepsilon)(s_1, t_1)^+ \varepsilon = u' s_1^+ R \\
(u, \varepsilon)(s_1, t_1)^+ (s_2, t_2)^+ \varepsilon = (u' \hat{z}, R)(s_1, \varepsilon)^+ (s_2, \varepsilon)^+ [(s_1, \varepsilon) + (s_2, \hat{z})]^+ \varepsilon
\]

Therefore \( \varepsilon = \text{lcs}(u', \hat{s}_1, \hat{z}_2, \hat{s}_2, \hat{s}_1) \). If \( \text{lcs}(u', \hat{s}_1) = \varepsilon \) then \( u s_1 t_1 \) would be a witness. If \( \text{lcs}(u', \hat{z}_2) = \varepsilon \) then \( u s_2 t_2 \) would be a witness. We therefore need to consider the cases \( \text{lcs}(u', \hat{s}_1) = \varepsilon \) and \( \text{lcs}(u', \hat{z}) = \varepsilon \). In fact, \( \text{lcs}(\hat{s}_1, \hat{z}) \neq \varepsilon \) holds as \( \hat{z} \hat{R} = \hat{z}_1 \hat{R} = \hat{t}_2 \hat{R} = R \) and therefore \( \hat{z} \subset \hat{s}_1 \). Thus \( \text{lcs}(u', \hat{s}_1) = \varepsilon \) if and only if \( \text{lcs}(u', \hat{z}) = \varepsilon \) and therefore if \( u s_1 s_2 t_2 t_1 \) is a witness if and only if \( u s_2 s_2 t_2 t_2 \) is a witness.
Thus, assume that $t_1 \neq \varepsilon$. Then $\hat{t}_1 \neq \varepsilon \neq \tilde{t}_1$ and therefore $\varepsilon \vdash \frac{1}{2} \lcs(u', \hat{t}_1, \tilde{t}_1)$. We obtain

$$
\begin{align*}
\lcs(L) &= \lcs\left(\begin{array}{c} u \\ us_1s_1t_1t_1 \\ us_2s_2t_2t_2 \end{array}\right) = \lcs\left(\begin{array}{c} u' R \\ u'\hat{t}_1\hat{s}_1\tilde{t}_1 R \\ u'\hat{z}\tilde{t}_1\hat{s}_2\tilde{t}_1 R \end{array}\right) \\
&= \lcs\left(\begin{array}{c} u \\ us_1s_2t_2t_1 \\ us_2s_1t_1t_2 \end{array}\right) = \lcs\left(\begin{array}{c} u' R \\ u'\hat{z}\tilde{t}_1\hat{s}_1\tilde{t}_1 R \\ u'\hat{z}\tilde{t}_1\hat{s}_2\tilde{t}_1 R \end{array}\right)
\end{align*}
$$

\begin{lemma}
Let $L = (u, \varepsilon)[(s_1, t_1) + (s_2, t_2)]^\ast w$ be wf.

Then $\lcs(L) = \lcs\left(\begin{array}{c} uw \\ us_1w_{t_1} \\ us_1s_1w_{t_1t_1} \end{array}\right) = \lcs\left(\begin{array}{c} uw \\ us_1w_{t_1} \\ us_1s_2w_{t_2t_1} \end{array}\right)$

If $s_1 = \varepsilon$, then $us_1w_{t_1t_1}$ is not required.
\end{lemma}

\begin{proof}
$R := \lcs(L)$.

Wlog. $s_i t_i \neq \varepsilon$.

Case $R \not\subseteq w = u' R$:

$w' \neq \varepsilon$

$\exists i : R t_i = \hat{t}_i R$

$R \not\subseteq us_iw_{t_i} = us_iu' R t_i$

$R \not\subseteq R t_i$

$L = (u, 1)[(s_1, t_1) + (s_2, t_2)]^\ast w' R = (u, R)[(s_1, \hat{t}_1) + (s_2, \tilde{t}_2)]^\ast w' \varepsilon \vdash \frac{1}{2} \lcs(u', w', \hat{t}_1, w', \tilde{t}_2)$

$\lcs(w', \hat{t}_1) = \varepsilon \lor \lcs(w', \tilde{t}_2) = \varepsilon$

$\lcs(L) = \lcs\left(\begin{array}{c} uw \\ us_1w_{t_1} \end{array}\right)$

Case $w \not\subseteq R = R' w$:

$\exists i : w t_i = \hat{t}_i w$

$R = R' w \not\subseteq us_iw_{t_i}$

$w \not\subseteq w t_i$

$L = (u' R', w)[(s_1, \hat{t}_1) + (s_2, \tilde{t}_2)]^\ast 1$

Apply Lemma to $L' = (u' R', 1)[(s_1, \hat{t}_1) + (s_2, \tilde{t}_2)]^\ast 1$

\end{proof}
Lemma 49. Let $L = (r, e)(s, t)^{k+1}$ be wf with $r \subseteq lcs(L)$ and $t \notin e \land rt = \text{ir} \land r \nsubseteq rs$ Then:
\[
\exists x, y, \forall k \geq 0: (r, e)(s, t)^{k+1} = (x, r)(\bar{s}, \bar{t})^ky \land lcs(L) = lcs(xy, \text{ir})r = lcs(st, ts)t \nsubseteq \text{ir}
\]
Proof.

If $x \nsubseteq e$ as $r \nsubseteq rs$

Thus also $r \nsubseteq rs = rp^m = q^m r$

Case $r \nsubseteq t = t'$:

$t' \neq e$

\[
i = t' \text{ as } rt't = rt = \text{ir}
\]

\[
r^{k+1}t^{k+1} = rs^{k+1}(t')^{k+1} = rsst't
\]

$L = (rs, r)(s, i)^{t'} = rst'r + (rss, ir)(s, i)^{t'}$ with $rstr = rst'$

$x := rs, y := t', \bar{s} := s$

With $r \nsubseteq rs$:

\[
lcs\left(\frac{rst}{rs^{k+2}t^{k+2}}\right) = lcs\left(\frac{rst'}{rs^{k+2}t^{k+2}t'r}r\right) = lcs\left(\frac{rs}{r}t'r \nsubseteq rt'r = \text{ir}\right)
\]

\[
lcs\left(\frac{rs}{r}t'r = lcs\left(\frac{rst'}{i}r\right)\right.
\]

\[
lcs(L) = lcs\left(\frac{rst}{rsstt}\right) = lcs\left(\frac{rst'}{rsst't}r\right) = lcs\left(\frac{rst'}{i}r\right) = lcs\left(\frac{xy}{i}r\right) \nsubseteq lcs(L)
\]

\[
lcs(st, ts)t = lcs(L) \text{ as}
\]

\[
lcs\left(\frac{rst}{rsstt}\right) = lcs\left(\frac{rst}{rsst't}r\right) = lcs\left(\frac{rs}{r}t'rs\right)t = lcs\left(\frac{ts}{st}t\right)
\]

Case $t \nsubseteq r = t'$:

$t't = t'$

\[
r'tt = rt = \text{ir} \land \text{ir} \land \text{ir}'t
\]

$x := t', y := e, \bar{s} := s$

With $r \nsubseteq rs$:

\[
i \nsubseteq l s
\]

\[
i' = r't \nsubseteq rs = r'ts = l\bar{s}r'
\]
Lemma 50. Let \( \text{lcs}(L) \) be the longest common subsequence of two strings. If \( \exists t \in \mathbb{R} \) such that \( \text{lcs}((l, s)) \supseteq \text{lcs}(L) \) then further \( \text{lcs}((l', s')) \supseteq \text{lcs}(L) \).

Proof.

Wlog. \( r_2 \subseteq r_1 = r_1' r_2 \):

\[
\begin{align*}
\text{wlog. } r_1' r_2 & \Rightarrow (s, \epsilon) [\exists t, \hat{t} \in \mathbb{R} \text{lcs}(L)] * w \\
\exists t_0: r_1 t_0 = \hat{t}_0 & \Rightarrow (s, \epsilon) [\exists t, \hat{t} \in \mathbb{R} \text{lcs}(L)] * w
\end{align*}
\]

Moving \( r_1 \) from \( w \) to the end using \( r_1 t_2 = \hat{t}_2 r_1 \) yields

\[ L' = (u, r_1)[(s, t_1) + (s, \hat{t}_2)] * w' \]
Apply lemma [18] on \((u, r_1)[(s_1, \epsilon) + (s_2, \hat{\ell}_2)]^* w'\).

Assume \(w' \subseteq r_1 w \wedge \hat{r}_1 \neq \epsilon\) from here on.

If \(w' \subseteq r_2 = \hat{r}_2 w: \)

\[
\hat{r}_1 = r'_1 \hat{r}_2 \\
\hat{r}_1 w = r_1 r_2 = r'_1 \hat{r}_2 w \\
u = u' \hat{r}_2 \\
\hat{r}_2 w = r_2 \subseteq R \subseteq uw
\]

Thus

\[
L = (u, \epsilon)[(s_1, r'_1 \hat{r}_2 w t_1 r'_2 w) + (s_2, \hat{r}_2 w t_2 \hat{r}_2 w)]^* w' \subseteq (u' \hat{r}_2, w)[(s_1, r'_1 \hat{r}_2 t_1 r'_2) + (s_2, \hat{r}_2 t_2 \hat{r}_2)]^* \epsilon
\]

Thus

\[
L = (u', r_2)[(\hat{s}_1, \hat{r}_1 t_1 r'_1) + (\hat{s}_2, t_2)]^* \epsilon
\]

This case is thus a special case of \(r_2 \subseteq w\) with \(r_2 = w = \epsilon\).

Assume \(r_2 \subseteq w = w' r_2\) from here on.

As

\[
= r_2(\hat{r}_1 t_1 r_1) = (\hat{r}_1 t_1 r'_1) r_2 \quad \text{and} \quad r_2(\hat{r}_2 t_2 r_2) = \hat{r}_2 t_2 r_2
\]

we can move \(r_2\) from \(w = w' r_2\) to the end of \(L\) s.t.

\[
L = (u, \epsilon)[(s_1, r'_1 \hat{r}_2 w t_1 r'_2 w) + (s_2, \hat{r}_2 w t_2 \hat{r}_2 w)]^* w' r_2 = (u, r_2)[(s_1, r'_1 t_1 r'_2) + (s_2, t_2)]^* w'
\]

Assume thus wlog. \(r_2 = \epsilon\) from here on s.t. \(w = w'\) and \(r_1 = r'_1\) and \(r_1 r_2 = \hat{r}_2 r_1\) and

\[
L = (u, \epsilon)[(s_1, r'_1 t_1 r_1) + (s_2, t_2)]^* w = (u, \epsilon)[(s_1, \hat{r}_1 w t_1 \hat{r}_1 w) + (s_2, t_2)]^* w
\]

If \(R \subseteq w:\)

Then:

\[
= w = w' R \wedge w' \neq \epsilon \\
= r_1 w = r_1 w' R \\
L = (u, \epsilon)[(s_1, r'_1 w' R t_1 \hat{r}_1 w' R) + (s_2, t_2)]^* w' R
\]

\[
\exists \ell_2: R \ell_2 = \ell_2 R \text{ as} \]

\[
R \subseteq us_2 w = us_2 w' R \text{ i.e. } R \subseteq R \ell_2
\]

\[
L = (u, R)[(s_1, \hat{r}_1 w' t_1 \hat{r}_1 w') + (s_2, t_2)]^* w'
\]

As \(w' \neq \epsilon\) we have \(R = \text{lcs}''(w' R, \ell_2 R)\)

\[
R = \text{lcs}''(L) = \text{lcs}''(uw, us_2 w')
\]
Assume \( w \not\subseteq R \) from here on.

W.l.o.g. \( w = \varepsilon \) as

\[ \exists \tilde{w}_2 : \tilde{w}_2 = \tilde{t}_2 w \]

\( w \not\subseteq R \subseteq \text{us}_2 \tilde{w}_2 \) i.e. \( w \not\subseteq \tilde{w}_2 \)

Thus

\[ L = (u, w)[(s_1, \tilde{t}_1 t_1 r_1) + (s_2, \tilde{t}_2)]^* \varepsilon \]

Hence:

\[ u = u' R \]
\[ r_1 = \tilde{t}_1 w = \tilde{r}_1 \]

\[ L = (u' R, \varepsilon)[(s_1, \tilde{t}_1 t_1 r_1) + (s_2, t_2)]^* \varepsilon \]

\( \text{us}_1 \not\subseteq \text{us}_1 \tilde{t}_1 r_1 \) as

\( \text{us}_1 \tilde{t}_1 r_1 \) is \( \text{wf} \)

\[ \exists \hat{s}_1 : r_1 s_1 = \hat{s}_1 r_1 \]

\( \text{us}_1 s_1 \tilde{t}_1 t_1 r_1 = \text{us}_1 \tilde{t}_1 s_1 \tilde{t}_1 t_1 r_1 \) is \( \text{wf} \)

\[ r_1 s_2 \tilde{t}_2 \tilde{t}_1 r_1 \) is \( \text{wf} \) for all \( l \) as

\( \text{us}_1 s_2 \tilde{t}_2 \tilde{t}_1 t_1 r_1 = \text{us}_1 \tilde{t}_1 s_2 \tilde{t}_2 \tilde{t}_1 t_1 r_1 \) is \( \text{wf} \) for all \( l \)

If \( R \subset r_1 \):

\[ r_1 = r_1' R \land r_1' \neq \varepsilon \]

If we have at least one copy of \( (s_1, \tilde{t}_1 t_1 r_1) \), then the word ends on \( r_1 = r_1' R \):

\[ (u, \varepsilon)[(s_1, \tilde{t}_1 t_1 r_1) + (s_2, t_2)]^* (s_1, \tilde{t}_1 t_1 r_1) [(s_1, \tilde{t}_1 t_1 r_1) + (s_2, t_2)]^* \varepsilon \]

\[ \not\subseteq (u, r_1' R)[(s_1, \tilde{t}_1 t_1 r_1) + (s_2, t_2)]^* (s_1, \tilde{t}_1 t_1) [(s_1, t_1) + (s_2, \tilde{t}_2)]^* \varepsilon \]

Thus by lemma \([47]\)

\[ R = \text{lcs}(u, s_1 s_2, u s_1 s_2 t_2) \]

In this case we might not be able to replace \( \text{us}_2 \tilde{w}_2 t_2 r_1 \) by \( \text{us}_2 s_1 w \tilde{t}_1 t_1 r_1 t_2 = \text{us}_2 s_1 w \tilde{t}_1 \tilde{t}_2 r_1 \)

or \( \text{us}_2 s_2 \tilde{t}_2 \tilde{t}_1 t_1 r_1 \).

Assume \( r_1 \not\subseteq R = R' r_1 \) from here on.

Thus:

\[ u = u' R' r_1 \]
\[ \exists \hat{t}_1 : R' t_1 = t_1 R' \]

\( R = R' r_1 \not\subseteq \text{us}_1 t_1 r_1 \) for all \( k \)

\[ L = (u' R', r_1, \varepsilon)[(s_1, \tilde{t}_1 t_1 r_1) + (s_2, t_2)]^* \varepsilon \]

If \( r_1 \not\subseteq r_1 \): \( \hat{s}_2 : r_1 s_2 = \hat{s}_2 r_1 \)

\[ L = (u' R', r_1)[(\hat{s}_1, t_1) + (\hat{s}_2, \tilde{t}_2)]^* \varepsilon \]

Apply lemma \([47]\)
Assume \( r_1 \not\leq r_1 s_2 \) from here on.

Thus (see also lemma \[10\])

\[- s_2 t_2 \not\leq t_2 s_2 \text{ as } r_1 t_2 = \hat{t}_2 r_1 \]

\[- \exists x_2, y_2, \tilde{s}_2 \forall l: r_1 s_2 t^{l+1} x_2 \tilde{s}_2 y_2 \tilde{t}_2 r_1 \]

\[- \text{lcs}(r_1, s_2) \vee (s_2, t_2) = \text{lcs}(s_2 t_2, t_2 s_2) t_2 \not\leq \hat{t}_2 r_1 \] with \(|x_2 y_2| = |s_2 t_2| \geq |t_2| = |t_2| \]

\[- R = R' r_1 \subseteq \text{lcs}(u, R' r_1, s_2) (s_2, t_2) + \vee \subseteq s_2 y_2, \hat{t}_2 r_1 \]

\[- \exists \tilde{t}_2: \tilde{t}_2 = \tilde{t}_2 R' \wedge \tilde{t}_2 \not\leq \vee \]

\[- \exists \tilde{z}_2: x_2 y_2 = \tilde{z}_2 R' \land \tilde{z}_2 \not\leq \vee \]

Partition \( L \) as follows:

1. \( u' \)
   \[ u = u' R \]

2. \( \text{as} \)
   \[ (u, \tilde{v})(s_1, \text{tr} t r_1) + \vee \]

3. \( \text{as} \)
   \[ (u, \tilde{v})(s_1, \text{tr} t r_1) + \vee \]

4. \( \text{as} \)
   \[ (u, \tilde{v})(s_1, \text{tr} t r_1) + \vee \]

5. \( \text{as} \)
   \[ (u, \tilde{v})(s_1, \text{tr} t r_1) + \vee \]

If \( R' \not\subseteq t_1 \):

\[- t_1 = t'_1 \text{tr} R' \not\subseteq t_1 \not\leq \vee \tilde{t}_1 = R' t'_1 \]

The partition of \( L \) thus becomes

1. \( u' R \)

2. \( \text{as} \)
   \[ (u' R', r_1)(\tilde{s}_1, t_1) + \vee = \ldots t'_1 R \]

3. \( \text{as} \)
   \[ (u' R', R)(\tilde{s}_1, t_1) + \vee = \ldots z_2 R \ldots \]

4. \( \text{as} \)
   \[ (u' R', R)(\tilde{s}_1, t_1) + \vee \]

5. \( \text{as} \)
   \[ (u, R)(\tilde{s}_1, t_1) + \vee \]

Hence \( R = \text{lcs}(u, R, t'_1 R, z_2 R, \tilde{t}_2 R) \) (as \( \tilde{t}_1 = R' t'_1 \))

\[ R = \text{lcs}^n(u, u s_1 \text{tr} t r_1, u s_2, u s_2 t_2) = \text{lcs}^n(u, u s_1 \text{tr} t r_1, u s_2, u s_2 t_2) \]

Assume \( t_1 \not\subseteq R' \) from here on s.t. \( R' t_1 = \hat{t}_1 R' \)

\[ \exists \tilde{s}_1: R' \tilde{s}_1 = \tilde{s}_1 R' \]

\[ R = R' t_1 r_1 \subseteq u s_1 \text{tr} t r_1 = u' R' t_1 r_1 = \tilde{s}_1 R' \tilde{s}_1 t_1 r_1 \]

i.e. \( R' \subseteq R' \tilde{s}_1 \)
Hence:

1. \[ u = u'R \]

2. \[ (u, \varepsilon)(s_1, t_1) \]

3. \[ (u, \varepsilon)(s_1, t_1) \]

4. \[ (u, \varepsilon)(s_1, t_1) \]

5. \[ (u, \varepsilon)(s_1, t_1) \]

If \( i_1 \neq \varepsilon \):

\[ R = \text{lcs}'(u, s_1, t_1, i_2, z_2, R, i_1, t_1, t_2) \]

If \( i_1 = \varepsilon \):

\[ R = \text{lcs}'(u, s_1, t_1, i_2, z_2, R, i_1, t_1, t_2) \]

Lemma 51. Let \( L = (u, \varepsilon)[(s_1, t_1) + (s_2, t_2)]^*w \) be wff with \( t_1 \neq \varepsilon \). Then

\[ \exists p, k_1,k_2, i_1, \ldots, i_j \in \{ 1, 2 \}^+ : u s_{i_1} \ldots s_{i_j} \bar{w} = u s_{i_1} \ldots s_{i_j} p^{k_1} \]

s.t. \[ L \equiv u(p^{k_1} + p^{k_2})^*w \]

Proof.

\[ R := \text{lcs}'(L) \]

\[ \exists p_i, \tilde{p}_i, m_i, n_i : s_i = p_i^{m_i} \land m_i \geq n_i \land s_i \bar{w} \]

If \( t_1 = \varepsilon \), then:

\[ \text{let } p_i, m_i \text{ s.t. } s_i = p_i^{m_i} \text{ and } p_i \text{ primitive; set } n_i := 0. \]

Assume thus \( t_1 \neq \varepsilon \). (Note that \( t_1 \neq \varepsilon \) already by assumption of the lemma.)

Then \( s_i^{k} \bar{w} = t_i^{k} \).

Hence by Lemma \[ 34 \]

For all \( i_1, \ldots, i_j \in \{ 1, 2 \}^+ \) we thus have

\[ u s_{i_1} \ldots s_{i_j} \bar{w} \]

If \( m_i = n_i + 1 \), then:
For all $i_1, \ldots, i_l \in \{1, 2\}^+$:
\[
us_1 \ldots u_i wr_1 r_1 i_1 \ldots r_{i-1} r_i = uw
\]
s.t. $L \not= uw$ with $\text{lcs}^p(L) = uw$.

Hence, $p$ can be chosen as $\varepsilon$.

So assume that $m_1 > n_1 \lor m_2 > n_2$.

We then have $p_1 = p_2 = p$.

If $s_2 = \varepsilon$, then:
\[
t_2 = \varepsilon \text{ as } L \text{ is } \text{wf and thus nonnegative.}
\]

Thus simply choose $p_2$ as $p_1$.

So assume $s_2 \neq \varepsilon$ and so $p_2 \neq \varepsilon \land m_2 > 0$.

Then for all $k, l \geq 1$ the following has to be $\text{wff}$:
\[
s_1^k s_2^l u \tau_1 r_1 r_2 \tau_1 \tau_2^k = p_1^{k m_1} p_2^{l (m_2-n_2)} \omega \tau_2 r_2 \tau_1 \tau_2^k
\]

If $m_2 > n_2$, then:
\[
p_2^l w = t_2^r r_1 = p_1^r w
\]

So assume $m_2 = n_2 > 0$ and thus $t_2 \neq \varepsilon$ and $m_1 > n_1$.

Then for all $k, l \geq 1$ the following has to be $\text{wff}$:
\[
s_2^k s_1^l u \tau_1 r_1 r_2 \tau_2^k = p_1^{k m_2} p_2^{l (m_1-n_1)} \omega \tau_1 r_1 \tau_2 \tau_2^k
\]

Hence: $p_1^r w = t_2^r r_2 = p_2^r w$

As $s_1, s_2 \in p^+$:
\[
L \not= u(p^{m_1-n_1} + p^{n_2-n_2})^* w
\]
\[
\text{lcs}^p(L) = \text{lcs}^p(uw, us_1 w \tau_1 r_1, us_2 w \tau_2 r_2)
\]

\begin{lemma}
Let $L = (u, \varepsilon)((s_1, r_1 i_1 r_1) + (s_2, r_2 t_2 r_2))^{*} w$ be $\text{wff}$ with $t_1 \neq \varepsilon$.

Then $\exists p, k_1, k_2 \forall i_1, \ldots, i_l \in \{1, 2\}^+$ s.t.
1. $us_1 \ldots s_i s_1 w \tau_1^r i_1 r_1 \not= us_1 \ldots s_i p^{k_1} w$
2. $us_1 \ldots s_i s_2 w \tau_2^r t_2 r_2 \not= us_1 \ldots s_i p^{k_2} w$
3. $L \not= u(p^{k_1} + p^{k_2})^* w$
4. $\text{lcs}^p(L) = \text{lcs}^p(s_1 w \tau_1 i_1 r_1, us_2 w \tau_2 t_2 r_2)$

\end{lemma}

\textbf{Proof.}

W.l.o.g. $t_2 \neq \varepsilon$ otherwise see Lemma 51

\begin{align*}
R := & \text{lcs}(L) \\
\exists p, \tilde{p}, m_1, n_1 : & s_1 = p^{m_1} \land t_1 = \tilde{p}^{n_1} \land m_1 \geq n_1 \land s_1 w \tau_1 i_1 r_1 = p^{m_1-n_1} w \tau_1
\\
s_1^k w \tau_1 \tau_2^k \text{ is wff for all } k \geq 1
\\
s_1^k w = t_1^r r_1 \text{ as } t_1 \neq \varepsilon
\end{align*}
W.l.o.g. \( p, \hat{p} \) primitive
\[
p^* w = \hat{p}^* r_1
\]
\( \exists \hat{p}, n_2: t_2 = \hat{p}^{n_2} \wedge \hat{p}r_2\bar{r}_1 \hat{p} \equiv r_2\bar{r}_1 \) as
\[
t_2^l r_2\bar{r}_1 \hat{p}^l = t_2^l r_2\bar{r}_1 \hat{p}^{n_1-k} \]
has to be \( \text{wwf} \) for all \( k, l \geq 1 \)
\[
t_2^l r_2 = \hat{p}^l r_1
\]
\( \hat{p} \) primitive as \( \hat{p} \) primitive
\( \exists m_2: s_2 = p^{m_2} \) as
If \( s_2 = \varepsilon \), set \( m_2 := 0 \).
So assume \( s_2 \neq \varepsilon \).
Then \( r_2 \supset s_2^* w \) as
\[
s_2^l w\bar{r}_2 \]
is \( \text{wfw} \) for \( l \) sufficiently large
So \( s_2^l w = \hat{p}^* r_1 \) as
\[
s_2^l w\bar{r}_2 t_2^l r_2\bar{r}_1 \]
is \( \text{wwf} \) for all \( k, l \geq 1 \)
Let \( c > n_2 \) and \( l \) so large that \( r_2 \supset s_2^l w \):
\[
s_2^l w\bar{r}_2 t_2^l r_2\bar{r}_1 \]
For all \( i_1 \ldots i_t \in \{1, 2\}^* \) we have:
- \( u_{s_1} \ldots s_i \) \( s_1 w \bar{r}_2 t_1\bar{r}_1 \)
- \( u_{s_1} \ldots s_i \) \( s_2 w\bar{r}_2 \)
Wlog. \( s_2 \neq \varepsilon \).
If \( w \supset r_2 = r_2^l \), then:
\( u_{s_1} \ldots s_i s_2 w \bar{r}_2 \)
is \( \text{wfw} \) for all \( i_1 \ldots i_t \).
Further \( pr_2 \bar{r}_2 \hat{p} \equiv r_2 \bar{r}_2 \) s.t.
\[
t_2^l r_2 \bar{r}_2 \equiv \hat{p}^{n_2} r_2 \bar{r}_2 \bar{r}_2 \]
Hence:
\[
u_{s_1} \ldots s_i \) \( s_2 w \bar{r}_2 t_2 r_2^{-1} \)
\( u_{s_1} \ldots s_i \) \( s_2 r_2^{-1} \)
So assume \( r_2 \supset w = w' r_2 \).
Then \( pw' = w' \hat{p} \) as:
\[
p w \bar{r}_2 \hat{p} \equiv p \bar{r}_2 \hat{p} \equiv w \bar{r}_2 \hat{p} \equiv \hat{p}
\]
Thus:
\[
u_{s_1} \ldots s_i \) \( s_2 w \bar{r}_2 t_2 r_2^{-1} \)
\( u_{s_1} \ldots s_i \) \( s_2 w' r_2^{-1} \)
\( u_{s_1} \ldots s_i \) \( s_2 p^{m_2+n_2} w' r_2^{-1} \)
\( u_{s_1} \ldots s_i \) \( s_2 p^{m_2+n_2} w' r_2^{-1} \)
\( u_{s_1} \ldots s_i \) \( s_2 p^{m_2+n_2} w' r_2^{-1} \)
Thus:
\[ L \equiv u(p^{m_1-n_1} + p^{m_2+n_2})^*_w \]
and \( \text{lcs}^*(L) = \text{lcs}(uw, upw) = \text{lcs}(uw, u_{s_1}w_{r_1}r_1, u_{s_2}w_{r_2}r_2) \)

Note that because of Lemma 53 the preceding results also apply to the \( \text{lcs}^\text{ext} \) for the respective languages.

### A.13 Lemma 30 in the main work

We first show that the computation of \( \text{lcs}^\text{ext} \) can be reduced to that of \( \text{lcs} \) which is essential to the proof of Lemma 30. Give an example on that and then prove Lemma 30.

#### Lemma 53. Let \( L \subseteq A^* \) with \( R = \text{lcs}(L) \). If \( \text{lcs}^\text{ext}(L) \in A^* \), then:

\[ \forall xR \in L \setminus \{ R \} \exists m \in \mathbb{N}: \text{lcs}(x^\omega L) = \text{lcs}^\text{ext}(L)^\omega x^m \]

**Proof.**

Fix any \( xR \in L \setminus \{ R \} \).

Thus \( x \not\equiv s \) s.t. there is some \( m \in \mathbb{N} \) with \( |x|m| > \text{lcs}(L) \).

As \( \text{lcs}^\text{ext}(L) \in A^* \) there is some \( y \in L \) s.t. for all \( zR \in L \):

\[ \text{lcs}^\text{ext}(L) = \text{lcs}(x^\omega, y^\omega) = \text{lcs}(x^\omega, y^\omega, z^\omega) = \text{lcs} \left( \frac{\text{lcs}(x^\omega, y^\omega)}{\text{lcs}(x^\omega, z^\omega)} \right)^\omega x^m \subseteq x^\omega \]

Pick any \( zR \in L \).

If \( \text{lcs}(x^\omega, z^\omega) \geq x^m \), then:

\[ \text{lcs}(x^\omega, z^\omega) \geq \text{lcs}(x^m, z^\omega) = x^m \]

If \( \text{lcs}(x^\omega, z^\omega) \geq x^m \), then:

\[ \text{lcs}(x^\omega, z^\omega) = \text{lcs}(x^m, z^\omega) = \text{lcs}(x^m, x^m z) \text{ (Lemma 34)} \]

In particular for \( z = y \) we thus have

\[ \text{lcs}^\text{ext}(L) = \text{lcs}(x^\omega, y^\omega) = \text{lcs}(x^m, x^m y) \]

Hence:

\[ \text{lcs}(x^m L) = \text{lcs}(x^m zR \mid zR \in L) = \text{lcs}(x^m xR, x^m xR, x^m zR \mid zR \in L) = \text{lcs}(x^m xR, x^m yR) = \text{lcs}(x^m, x^m y) \]

#### Example 54. In the case of \( L = \{ s^k t^k R \mid k \in \mathbb{N}_0 \} \) with \( R = \text{lcs}(L) \) we have \( \text{lcs}^\text{ext}(L) = \text{lcs}( (st)^{\omega} (s^{k+2} t^{k+2})^{\omega} \mid k \geq 0) \). If \( s \) and \( t \) commute, then \( \text{lcs}^\text{ext}(L) = (st)^{\omega} \), and \( \text{lcs}^\text{ext}(L) = (st)^{\omega} \) is unbounded. Thus assume \( st \not\equiv ts \) s.t. \( \text{lcs}(ts, st^{k+1}) = \text{lcs}(ts, st) \) and

\[ \text{lcs}( (st)^{\omega}, (s^{k+2} t^{k+2})^{\omega} ) = \text{lcs}( (ts)^{\omega} ts, (st^{k+2} s^{k+1})^{\omega} st^{k+1} ) = \text{lcs}(ts, st) \]

Hence, \( \text{lcs}(ts tl) = \text{lcs}(ts, st) R = \text{lcs}^\text{ext}(L) lcs(L) \).
Lemma 55 (Lemma \[30\] in the main work). Let \( G \) be a context-free grammar with \( N \) nonterminals and \( L := L(G) \subseteq B^* \) w.f. For every nonterminal \( X \) let \( r_X \in A^* \) s.t. \( |r_X| = d_X \) and \( r_X L_X \) w.f.

Then \( r_X L_X \approx_{ics} r_X L_X^{\leq 4N} \approx_{ics} T_{ics}^n(r_X L_X^{\leq 4N}) \).

Proof.

Let \( S \) be the axiom of \( G \).

Wlog. \( G \) is reduced to the nonterminals which are reachable from \( S \) and which are productive.

As \( r_X L_X \) is w.f., we have for any \( \zeta \in L_X \) that \( \rho(\zeta) = \pi v \in A^* \) with \( u \trianglelefteq r_X = r'_X u \) s.t. \( r_X \zeta \leq r'_X v \) and thus:

\[
|\rho(r_X \zeta)| = |r'_X v| = |r_X v| - |u| = \Delta(r_X \zeta)
\]

Let \( \kappa_0 \in L_X \) be a shortest-word-after-reduction i.e. \( \Delta(r_X \kappa_0) = \min\{\Delta(r_X \zeta) \mid \zeta \in r_X L_X\} \).

Let \( \kappa_1 \in L_X \) be a second-shortest-word-after-reduction (if it exists) i.e. \( \Delta(r_X \kappa_1) = \min\{\Delta(r_X \zeta) \mid \zeta \in r_X L_X \mid \Delta(\zeta) > \Delta(\kappa_0)\} \).

Then wlog. \( \kappa_0 \in L_X^{\leq N} \) and \( \kappa_1 \in L_X^{\leq 4N} \) as:

For any \( S \rightarrow_\circ \alpha X \beta \) and \( X \rightarrow_\circ \gamma \) we need to have that

\[
\forall k \geq 0: \Delta((\alpha, \beta)(\sigma, \tau)^k \gamma) = \Delta(\alpha \gamma \beta) + k \Delta(\sigma \tau) \geq 0.
\]

In fact, this has to hold also for any prefix of a word of \( L \) and \( r_X L_X \) s.t. also

\[
\forall k \geq 0: \Delta(\alpha \sigma^k \gamma) = \Delta(\alpha \gamma) + k \Delta(\sigma) \geq 0
\]

and thus \( \Delta(\tau) \geq -\Delta(\sigma) \).

Any word \( \zeta \in L_X \setminus L_X^{\leq N} \) has a derivation tree with a path from its root to some leaf along which at least \( N + 1 \) nonterminals occur, i.e. along at least one nonterminal occurs twice which gives rise to a factorization of the form

\[
\zeta = (\alpha, \beta)(\sigma, \tau)^\gamma
\]

s.t.

\[
|\rho(\zeta)| = \Delta(\zeta) = \Delta(\alpha \gamma \beta) + \Delta(\sigma \tau) \geq \Delta(\alpha \gamma \beta) = |\rho(\alpha \beta \gamma)|
\]

Removing the pumping tree that gives rise to the factor \( (\sigma, \tau) \) thus leads to a word \( \alpha \gamma \beta \) that is shorter than \( \zeta \) before reduction, and at most as long as \( \zeta \) after reduction.

Hence, \( r_X L_X^{\leq N} \) already contains all shortest-words-after-reduction, i.e.

\[
\min\{\Delta(r_X \zeta) \mid \zeta \in r_X L_X\} = \min\{\Delta(r_X \zeta) \mid \zeta \in r_X L_X^{\leq N}\}
\]

and thus wlog. \( \kappa_0 \in L_X^{\leq N} \).

Assume there exists a second-shortest-word-after-reduction \( \kappa_1 \in L_X \).

Any path that consists of at least \( 2N + 1 \) nonterminals contains at least one terminal three times which gives rise to a factorization of the form

\[
\kappa_1 = (\alpha, \beta)(\sigma_1, \tau_1)(\sigma_2, \tau_2)^\gamma
\]

If \( \Delta(\sigma_1 \tau_1) = \Delta(\sigma_2 \tau_2) \), we can prune both pumping trees.

So assume \( \Delta(\sigma_1 \tau_1) > 0 \) for either \( i = 1 \) or \( i = 2 \).

Pruning \( (\sigma_i, \tau_i) \) leads to \( (\alpha, \beta)(\sigma_j, \tau_j)^\gamma \) \( (j \neq i) \) with

\[
\Delta(\kappa_1) > \Delta((\alpha, \beta)(\sigma_j, \tau_j)^\gamma) \geq \Delta(\kappa_0)
\]
As \( \kappa_1 \) is a second-shortest word after reduction, we have to have
\[
\Delta((\alpha, \beta)(\sigma, \tau)) = \Delta(\kappa_0)
\]
and thus
\[
\Delta(\sigma, \tau) = 0
\]
s.t. we can prune \((\sigma, \tau)\) to obtain
\[
\Delta(\kappa_1) = \Delta((\alpha, \beta)(\sigma, \tau))
\]
a possible different second-shortest-word-after-reduction.

Let \( R := \text{lcs}'(r_X L_X) \).

If the \( \text{lcs}' \) of \( r_X L_X \) can be extended infinitely:

Then \( \rho(\kappa_0) = R \) and \( \rho(\kappa_1) = xR \) for some \( x \in A^+ \).

Then for any \( \zeta \in L_X \) we have \( \rho(r_X \zeta) = yR \) with \( x^\omega \partial y^\omega \), i.e. \( xy = yx \) and as \( |y| \geq |x| > 0 \) in fact \( y \in x^+ \).

Thus the test set of \( r_X L_X \) is in this case given by \( \{\rho(r_X \kappa_0), \rho(r_X \kappa_1)\} \subseteq r_X L_X^{2N} \).

Assume thus w.l.o.g. that the \( \text{lcs}' \) of \( r_X L_X \) can at most be finitely extended.

This implies that \( \rho(r_X L_X) \) contains at least three distinct words.

We distinguish the two cases whether \( R = \text{lcs}'(r_X L_X) \) is a strict suffix of every word in \( r_X L_X \), in particular \( R \subset \rho(r_X \kappa_0) \), or if \( R \) is a, and thus the shortest-word-after-reduction, in particular \( R = \rho(r_X \kappa_0) \).

\( \text{a} \) If \( R = \text{lcs}'(r_X L_X) \subsetneq \rho(r_X \kappa_0) \), there is some witness \( \kappa \in L_X \) s.t.
\[
R = \text{lcs}(\rho(r_X \kappa_0), \rho(r_X \kappa)) \supseteq r_X \kappa_0
\]
In particular, we have that \( r_X \kappa_0 \partial aR \) and \( r_X \kappa \partial bR \) for two distinct opening parenthesis \( a, b \in A \) (\( a \neq b \)).

\( \text{b} \) If \( R = lcs'(r_X L_X) = \rho(r_X \kappa_0) \), then recall Lemma [53].

The maximal extension \( E \) of the \( R \) is given by
\[
E = \text{lcsext}'(L) = \text{lcs}(\rho(r_X \zeta \bar{R})^\omega | \zeta \in L_X, \rho(r_X \zeta) \neq R)
\]
As \( E \) is assumed to be finite, \( \rho(r_X L_X) \) has to contain at least two other reduced words, both longer than \( \rho(r_X \kappa_0) \). In particular, there has to be a second-shortest-word-after-reduction \( \kappa_1 \) s.t. we find a witness \( \kappa \) for \( E \) w.r.t. \( \kappa_1 \):
\[
E = \text{lcs}(\rho(r_X \zeta \bar{R})^\omega | \zeta \in L_X, \rho(r_X \zeta) \neq R) = \text{lcs}(\rho(r_X \kappa_1 \bar{R})^\omega, \rho(r_X \kappa \bar{R})^\omega)
\]

Let \( \rho(r_X \kappa_1) = xR \land x \neq \varepsilon \) with \( x \neq \varepsilon \). Choose \( m > 0 \) s.t. \( E \subseteq x^m \).

Pick any \( \zeta \in L_X \) with \( \rho(r_X \zeta) = zR \land z \neq \varepsilon \) and w.l.o.g. \( xz \neq xz \).

We then have
\[
\text{lcs}(x^\omega, z^\omega) = \text{lcs}(xz, zw)
\]
If \( \text{lcs}(x^\omega, z^\omega) \subsetneq x^m \), then
\[
\text{lcs}'(x^\omega, z^\omega) = \text{lcs}'(x^m, x^m z) \subsetneq x^m
\]
If otherwise \( \text{lcs}(x^\omega, z^\omega) \supseteq x^m \), then
\[
\text{lcs}'(x^\omega, z^\omega) \supsetneq \text{lcs}'(x^m, x^m z) = x^m
\]
Thus,
\[
ER = \text{lcs}^\rho(x^m r_X L_X) \supseteq \rho(x^m r_X k_0) = x^m R = x^{m-1} \rho(r_X k_1)
\]
i.e. we can reduce this case to the case where \( R \) is a strict suffix of any word in \( r_X L_X \) by extending \( r_X \) to \( x^m r_X \).

Note that then any witness for \( ER \) w.r.t. \( x^m r_X k_0 \) is also a witness w.r.t. \( x^m r_X k_1 \) and vice versa.

Assume thus wlog. that \( R = \text{lcs}^\rho(r_X L_X) \supseteq \rho(r_X k_0) \) from here on.

Choose \( \kappa \) in \( L_X \) s.t.
1. \( \text{lcs}(\rho(r_X k_0), \rho(r_X k)) = \text{lcs}^\rho(r_X L_X) \).
2. \( |\kappa| \) is minimal w.r.t. to all words in \( r_X L_X \) satisfying 1.
3. \( |\rho(r_X k)| \) is minimal w.r.t. to all words in \( r_X L_X \) satisfying 2.

Wlog. \( \rho(r_X k_0) = \ldots aR \) and \( \rho(r_X k) = \ldots bR \) with \( a \neq b \) and \( a, b \in A \).

Note that there is a unique factorization \( r_X k = \zeta b \xi \) s.t. both \( \zeta \) and \( \xi \) are \( \text{wf} \) and \( \rho(\xi) = R \):

For every prefix (before reduction) \( \pi \) of \( r_X k \) we can interpret \( \Delta(\pi) \) as the height of the last letter of \( \pi \).

Then the \( b \) in \( \rho(r_X k) = \ldots bR \) is the last letter in \( r_X k \) of height \( \Delta(r_X k) - |R| \) and is, thus, uniquely identified.

This specific \( b \) splits \( r_X k \) into \( r_X k = \zeta b \xi \); as this \( b \) is the last letter in \( \kappa \) on height \( \Delta(\kappa) - |R| \), \( \xi \) has to be \( \text{wf} \) with \( \rho(\xi) = R \); as \( r_X k \) is \( \text{wf} \) and \( \zeta \) is a prefix thereof, trivially also \( \zeta \) is \( \text{wf} \).

Assume every derivation tree of \( \kappa \) contains a path to a letter within \( b \xi \) along which some nonterminal \( A \) occurs at least 4 times (see Fig. 1).

Note that \( b \) might be contained in \( r_X k \) s.t. \( \rho(\kappa) \subseteq \rho(\xi) = R \); specifically in the case where \( R \) was originally finitely extendable by some \( E \neq \varepsilon \).

This gives rise to a factorization
\[
\kappa = (\alpha, \beta)(\sigma_1, \tau_1)(\sigma_2, \tau_2)(\sigma_3, \tau_3)\gamma
\]
Any word
\[
\zeta \in \{(\alpha, \beta)(\sigma_1, \tau_1)^{k_1}(\sigma_2, \tau_2)^{k_2}(\sigma_3, \tau_3)^{k_3}\gamma \ | \ k_1 \in \{0, 1\}, k_2 + k_3 < 3\}
\]
is shorter (before reduction) than \( \kappa \), hence cannot be a witness w.r.t. \( k_0 \) i.e. \( aR \subseteq \text{lcs}^\rho(r_X k, r_X \zeta) \).

Let
\[
L = (r_X \alpha, \beta)[(\sigma_1, \tau_1) + \ldots + (\sigma_3, \tau_3)]^\gamma
\]
\( L \) is \( \text{wf} \) with \( R = \text{lcs}^\rho(r_X L_X) = \text{lcs}^\rho(L) \) as \( L \subseteq r_X L_X \) contains both \( r_X k \) and \( r_X \alpha \gamma \beta \) with the latter not a witness w.r.t. to \( r_X k_0 \) s.t. \( \text{lcs}^\rho(r_X k, r_X \alpha \gamma \beta) = R \).

Our goal is to show that already \( r_X \alpha \sigma_i \gamma_1 \tau_i \beta \) or \( r_X \alpha \sigma_i \gamma_2 \tau_j \beta \) for some \( i \neq j \) is a witness w.r.t. \( r_X \alpha \beta \).

Note that:
- \( r_X \alpha \) has to be \( \text{wf} \), all other factors \( \beta, \gamma, \sigma, \tau \) have to be \( \text{wwf} \).
- As note already at the beginning, we have both \( \Delta(\sigma) \geq 0 \) and \( \Delta(\tau) \geq 0 \).
- By choice of the path used for the factorization, we have \( \rho(\tau_1) \supseteq R = \text{lcs}^\rho(L') = \text{lcs}^\rho(r_X L_X) \).

We first reduce the factors \( \alpha, \beta, \gamma, \sigma, \tau \), to words in \( A^* \).
As \( r_X \alpha \) has to be \( \text{wf} \), simply set \( u := \rho(r_X \alpha) \in A^* \).
Wlog. we may assume \( \beta = \varepsilon \).
This amounts to changing $R = \text{lcs}^s(L)$ to $R := \rho(\text{lcs}^s(L)|J)$.  
Wlog, we may also assume $\rho(\sigma_i) = s_i \in A^*$:

Let $\rho(\sigma_i) = \tau_i y_i$ for any $i \in [3]$.  
Then $u\tau_i$ has to be $wf$ for all $i \in [3]$, i.e. we have $u \overset{\rho}{=} u\tau_i x_i$ for all $i \in [3]$.

As $\Delta(\sigma_i) \geq 0$, we have $\tau_i \overset{\rho}{\subseteq} \tau_i y_i$.

As $u\tau_i, \sigma_i \overset{\rho}{=} u\tau_i y_i y_i$ has to be $w$, we have $y_i = s_i x_i$.

Pick $J \in [3]$ s.t. $x_J$ is a longest word of $\{x_1, x_2, x_3\}$.  
Then $u\tau_i, \sigma_j \overset{\rho}{=} u\tau_i x_j \tau_i x_i, x_j x_j x_j$ has to be $w$, i.e. $(x_j \tau_i) s_i x_j x_j x_j$ is $w$ for all $i \in [3]$.

Thus there exist $s_i$ s.t. $p(x_j \tau_i) s_i = \hat{s}_i \rho(x_j \tau_i)$ resp. $x_j \tau_i s_i = \hat{s}_i x_j \tau_i$.

So $u\sigma_i \cdots \sigma_i \overset{\rho}{=} u\tau_i \hat{s}_i \cdots \hat{s}_i x_i x_j x_j x_j$.

Thus, set $u := \rho(u\tau_i)$, $\gamma := \rho(x_j \gamma)$ and $\sigma := \hat{s}_i$.

Analogously, we may further assume $\rho(\gamma) = w \in A^*$:

If $\rho(\gamma) = \tau w$, then $u = u' x$ resp. $u \overset{\rho}{=} u\tau x$ and thus $u \overset{\rho}{=} u\tau y_i \tau w$.

So $x s_i \tau$ is $w$ for all $i \in [3]$.

Hence, we find $\hat{s}_i$ with $x s_i = \hat{s}_i x$ s.t. $u s_i \gamma \overset{\rho}{=} u\tau \hat{s}_i \cdots \hat{s}_i x_i x_j x_j x_j$.

Thus, set $s_i := \hat{s}_i$, $u := \rho(u\tau)$ and $\gamma := w$.

We thus may simply assume that

$L = (u, 1)[(s_1, \tau_1) + \ldots + (s_3, \tau_3)]^* w$

with $L$ $w$ and $\rho(\tau_3 \tau_2 \tau_1) = \tau y$ with $y \overset{\rho}{=} R = \text{lcs}^s(L') = \text{lcs}^s(r_X L X)$.

For $i \in [3]$ we have either $\rho(\tau_i) = \tau r_i x_i$ or $\rho(\tau_i) = \tau r_i \tau r_i \wedge r_i \neq \varepsilon$.

Let $\rho(\tau_i) = \tau_i y_i$ for $i \in [3]$.

Then $L_i := (u, \varepsilon)(s_i, \tau_i)^* w$ has to be $w$ for any $i \in [3]$ as $L_i \subseteq L$ and $L$ is $w$ by assumption.

Thus $\tau_i \overset{\rho}{=} \tau_i y_i \tau_i y_i$ has to be $w$.

If $|y_i| \geq |x_i|$, then $y_i \tau_i$ has to be $w$, i.e. $x_i \overset{\rho}{=} \tau_i y_i$. Setting $r_i := x_i$ and $t_i := \rho(y_i \tau_i)$, we have

$\tau_i \overset{\rho}{=} \tau_i y_i = \tau_i \rho(y_i \tau_i) x_i = \tau_i t_i r_i$

Otherwise $|y_i| < |x_i|$ and $x_i \tau_i$ is $w$ with $y_i \overset{\rho}{=} \tau_i x_i$. Then set $r_i := y_i$ and $t_i := \rho(x_i \tau_i) \neq \varepsilon$ s.t.

$\tau_i \overset{\rho}{=} \tau_i y_i = \rho(x_i \tau_i) y_i y_i = \tau_i t_i r_i = \tau_i \tau_i t_i r_i$

Assume that for some $i \in [3]$ we have $\rho(\tau_i) = \tau_i \tau_i r_i$ with $t_i \neq \varepsilon$.

As shown in lemma [31] and lemma [32] we always have for $j \neq i$ and any sequence $i_1 \ldots i_k \in \{1, 2\}^+$:

$\rho u s_{i_1} \cdots s_{i_j} w_{\tau_i} r_j \overset{\rho}{=} u s_{i_1} \cdots s_{i_j} p^{m_{j}+n_{j}} w$  
$\rho u s_{i_1} \cdots s_{i_j} w_{\tau_i} r_j \overset{\rho}{=} u s_{i_1} \cdots s_{i_j} p^{-m_{j}-n_{j}} w$

Hence $L \overset{\rho}{=} u(p^{m_{1}+n_{1}} + p^{m_{2}+n_{2}} + p^{m_{3}+n_{3}})^* w$ and thus

$\text{lcs}^s(L) = \text{lcs}^s(uw, us_1 w \tau_1, us_2 w \tau_2, us_3 w \tau_3)$

So it remains the case that for all $i \in [3]$ we have $r_i = \tau_i t_i$.
Hence $L = (u, 1)[(s_1, \overline{r}_1 t_1 r_1) + (s_2, \overline{r}_2 t_2 r_2) + (s_3, \overline{r}_3 t_3 r_3)]^* w$.

As before $r_i \overline{r}_i$ has to be wwf for any $i, j \in [3]$, let $\{i_1, i_2, i_3\} = [3]$ s.t. $r_{i_3} \preceq r_{i_2} \preceq r_{i_1}$.

Then $t_{i_3} r_{i_3} \overline{r}_i$ is wwf for all $i \in [3]$ s.t. $\exists i_{i_3,i}: r_i \overline{r}_i t_{i_3} \preceq t_{i_3,i}, r_i r_{i_3}$.

Note that $\rho(t_{i_3} r_{i_3} \overline{r}_i t_2 r_2 \overline{r}_i t_1 r_1) = \overline{y}$ with $y \subseteq \text{lcs}(L) = R$.

Hence $|R| \geq |y| = \Delta(t_{i_3} r_{i_3} \overline{r}_i t_2 r_2 \overline{r}_i t_1 r_1) + |x| = |t_{i_3} t_2 r_2| + |r_3| + |x|$.

We show that $|R| \geq |y| \geq |t_i r_i|$ for all $i \in [3]$:

- If $|r_1| \leq |t_2 r_2| \wedge |r_2| \leq |t_3 r_3|$, then $|y| = |t_1 t_2 t_3 r_3| \geq |t_1 t_2 r_2| \geq |t_1 r_1|$
- If $|r_1| > |t_2 r_2| \wedge |r_2| + |r_1| - |t_2 r_2| \leq |t_3 r_3|$
  then $|t_2 r_2| < |r_1| \wedge |r_1| \leq |t_2 t_3 r_3|$
  and $|y| = |t_1 t_2 t_3 r_3| \geq |t_1 r_1| > |t_1 t_2 r_2|$
- If $|r_1| \leq |t_2 r_2| \wedge |r_2| > |t_3 r_3|$, then $|x| = |r_2| - |t_3 r_3|$
  and $|y| = |t_1 t_2 t_3 r_3| + |r_2| - |t_3 r_3| = |t_1 t_2 r_2| \geq \max(|t_1 r_1|, |t_1 t_2 t_3 r_3|)$
- If $|r_1| > |t_2 r_2| \wedge |t_3 r_3| < |r_2| + |r_1| - |t_2 r_2|$, then $|x| = |r_1| - |t_2 t_3 r_3|$
  and $|y| = |t_1 t_2 t_3 r_3| + |r_1| - |t_2 t_3 r_3| = |t_1 r_1| \geq \max(|t_1 t_2 r_2|, |t_1 t_2 t_3 r_3|)$

Consider $L' = (u, 1)[(s_1, \overline{r}_1 t_1 r_1) + (s_2, \overline{r}_2 t_2 r_2)]^*(s_3, \overline{r}_3 t_3 r_3) w$

We have $L' \subseteq L$ with $\kappa = (u, 1)(s_1, \overline{r}_1 t_1 r_1)(s_2, \overline{r}_2 t_2 r_2)(s_3, \overline{r}_3 t_3 r_3) w \in L'$ and thus $R = \text{lcs}(L) \subseteq \text{lcs}(L')$.

Note that $(u, 1)(s_3, \overline{r}_3 t_3 r_3) w$ cannot be a witness w.r.t. $uw$ as its length after reduction is strictly smaller than that of $r_X \kappa$, hence the two words have to coincide on at least the last 1 + |R| letters s.t. $\text{lcs}(L') \preceq \text{lcs}(r_X \kappa, (u, 1)(s_3, \overline{r}_3 t_3 r_3) w) = R$ i.e. $\text{lcs}(L) = \text{lcs}(L') = R$.

Let $\tilde{w} = \rho(s_3, \overline{r}_3 t_3 r_3) w$.

If $\tilde{w} \not\subseteq \overline{y}$ is only wwf, then $u = u' x$ and suitable conjugates of $s_i$ exist that allow us to move $x$ from $u = u' x$ through any sequence $s_{i_1} \ldots s_{i_k}$ next to $\tilde{w}$ as done before.

Thus assume wlog. that $\tilde{w}$ is already wff.

By lemma [50] we have:

$$R = \text{lcs}(L') = \text{lcs}(u \tilde{w}) = \text{lcs}(u s_1 \overline{r}_1 t_1 r_1, u s_2 \overline{r}_2 t_2 r_2, u s_3 s_1 \overline{r}_1 t_1 r_1, u s_2 s_2 \overline{r}_2 t_2 r_2)$$

Neither $u s_1 \overline{r}_1 t_1 r_1$ nor $u s_2 \overline{r}_2 t_2 r_2$ can be witnesses again because their length before reduction is strictly less than that of $r_X \kappa$.

Hence, either $u s_1 s_1 \overline{r}_1 t_1 t_1 r_1$ or $u s_2 s_2 \overline{r}_2 t_2 t_2 r_2$ is a witness w.r.t. $u s_3 w \overline{r}_3 t_3 r_3$ and thus also w.r.t. $uw$.

Wlog. $u s_1 s_1 \overline{r}_1 t_1 t_1 r_1$ is a witness.

Consider then $L'' = (u, 1)[(s_1, \overline{r}_1 t_1 r_1)^* + (s_3, \overline{r}_3 t_3 r_3)^*] w$.

Again $L'' \subseteq L$ s.t. $R = \text{lcs}(L) \subseteq \text{lcs}(L'')$.

But also $\text{lcs}(L'') \subseteq \text{lcs}(uw, u s_1 s_1 \overline{r}_1 t_1 t_1 r_1) = R$ s.t. $R = \text{lcs}(L) = \text{lcs}(L'')$.
Assume that the \( \text{lcse} \)-defining letter \( b \) is contained within \( \kappa \) and not \( r_X \), i.e. that \( \rho(\kappa) = \ldots b \text{lcse}(L) \) with \( r_X \kappa \text{ wf} \). As any opening letter within \( \kappa \) can only be canceled by a closing letter from right, by canceling out always the pair of matching opening and closing letters that is farthest to the right, we obtain a unique factorization of \( \kappa = \zeta b \xi \) s.t. \( r_X \zeta \) and \( \xi \) are both \( \text{wf} \) with \( \rho(r_X \kappa) = \rho(r_X \zeta)b \rho(\xi) = \rho(r_X \zeta)b \text{lcse}(L) \) s.t. this specific occurrence of the letter \( b \) defines the reduced suffix \( \text{lcse}'(L) \). (If \( b \) is contained within \( r_X \), then \( \kappa = \xi \), \( r_X = r_X' b \zeta \) and \( \rho(x \kappa) = \text{lcse}'(L) \).

We assume that the given derivation tree of the witness \( \kappa \) contains a path (drawn as dashed line) which (i) leads to one of the letters within \( b \zeta \) and (ii) consists of at least \( 3N + 1 \) nonterminals so that by the pigeon-hole principle at least one nonterminal \( A \) occurs at least 4 times; specifically, consider precisely the first \( 3N + 1 \) nonterminals along such path and let \( A \) be the nonterminal that occurs both at least 4 times within this fragment and also occurs the earliest. W.r.t. the nonterminal \( A \) we factorize the witness as \( \kappa = (\alpha, \beta)(\sigma_1, \tau_1)(\sigma_2, \tau_2)(\sigma_3, \tau_3)\gamma = \ldots b \text{lcse}'(L) \).

Using lemma \ref{lem:terminals3} and now that \( r_1 \not\subseteq R = \text{lcse}'(L') \), we obtain

\[
R = \text{lcse}'(L') = \text{lcse}' \left( \begin{array}{c}
us_1w\overline{t_1}t_1r_1 \\
us_3w\overline{t_3}t_3r_3 \\
us_1s_1w\overline{t_3}t_1t_1r_1 \\
us_1s_1w\overline{t_3}t_1t_1t_3t_3
\end{array} \right)
\]

But by our assumption that \( r_X \kappa \) is a witness w.r.t. \( uw \) of minimal length before reduction, none of these words can be witnesses. Hence, our assumption that such a factorization exists, cannot hold.

So, every path leading to the occurrence of \( b \) that defines the \( \text{lcse}' \) of \( L \) or to a letter right of it has to have height at most \( 3N \). By minimality, we can also assume that any path fragment that leads from the main path (leading to \( \text{lcse}' \)-defining occurrence of \( b \)) to a letter left of this \( b \) contains any nonterminal at most once (see Fig. \ref{fig:balancedness}). Hence, the derivation tree can have height at most \( 4N \).

\section*{A.14 Lemma \ref{lem:terminals3} in the main work}

\textbf{Lemma \ref{lem:terminals3} (Lemma \ref{lem:terminals3} in the main work).} If \( L = L(G) \) is not \( \text{wf} \), then there is some least \( h_0 \) s.t. \( r_X L_X^{h_0} \not\approx \) is not \( \text{wf} \) with \( X \rightarrow_G Y Z \). For \( h \leq h_0 \), all \( r_X L_X^{\leq h} \) are \( \text{wf} \) s.t. \( T_X^{\leq(h+1)} \approx_{\text{lcse}} \rho(r_X L_X^{\leq h}) \). If \( h_0 \geq 4N + 1 \), then at least for one nonterminal \( X \) we have \( T_X^{4N+1} \not\approx_{\text{lcse}} T_X^{4N} \).

\textbf{Proof.}
Assume that the \( lcs^\rho \)-defining occurrence of \( b \) is not contained in \( r_X \) s.t. \( \kappa = \zeta \xi \) with both \( r_X \zeta \) and \( \xi \text{wf} \) and \( \rho(r_X \zeta) = \rho(r_X \zeta) \text{b} \text{lcs}^\rho(L) \). Consider any path that leads to a letter within \( \zeta \). (If \( b \) is contained in \( r_X \), then this cannot happen.) The first nonterminal along this path that is not also contained in the path leading to \( b \) defines a subtree that does not contain the marked \( b \) anymore. Assume this subtree contains a path with at least \( N + 1 \) nonterminals s.t. we can factorize \( \zeta = (\mu, \nu)(\phi, \psi) \rho \). Then \( \kappa = (\mu, \nu \xi_3)(\phi, \psi) \rho \), and \( L' = (\mu, \nu \xi_3)(\phi, \psi) \rho \) is a sublanguage of \( L \) and thus \( r_X L' \) is \( \text{wf} \). Hence, \( (r_X \mu, \nu \xi_3)(\phi, \psi) \rho = r_X \mu \nu \xi_3 \) is \( \text{wf} \). As \( \xi \) is \( \text{wf} \), too, we have that \( r_X \mu \nu \xi_3 \) is a shorter (before reduction) witness than \( \kappa \). Hence, we can always assume that all subtrees rooted at a node left of the path leading to the marked \( b \) have height at most \( n - 1 \). Thus, if all paths leading to a letter within \( \xi \) contain at most \( 3N \) nonterminals, then the derivation tree can have at most height \( 4N \).

We write \( T_{\text{ia}}(r_X L_X^{\leq h}) \) for \( T_{\text{ia}}(\rho(r_X L_X^{\leq h})) \).

For simplicity, we also assume that all linear rules have been removed.

If \( G \) is \( \text{wf} \), then inductively we have \( T_X^{\leq h} \approx_{\text{ia}} r_X L_X^{\leq h} \) s.t.:

\[
\begin{align*}
T_X^{\leq h+1} &= T_X^{\leq h} \cup \bigcup_{X \rightarrow Z} T_Y^{\leq h} T_Z^{\leq h} \\
&= T_X^{\leq h} \cup \bigcup_{X \rightarrow Z} \rho(r_X r_Y) T_Y^{\leq h} T_Z^{\leq h} \\
&\approx_{\text{ia}} T_{\text{ia}}(T_X^{\leq h}) \cup \bigcup_{X \rightarrow Z} \rho(r_X r_Y) T_Y^{\leq h} T_Z^{\leq h} \\
&\approx_{\text{ia}} T_{\text{ia}}(T_X^{\leq h}) \cup \bigcup_{X \rightarrow Z} \rho(r_X r_Y) T_Y^{\leq h} T_Z^{\leq h} =: T_X^{\leq h+1}
\end{align*}
\]

If \( G \) is \( \text{wf} \), all \( r_X \) and \( T_X^{\leq h} \) can be computed for every \( h \) in polynomial time using SLPs. Further, \( \text{lcs}^\rho(r_X L_X^{\leq 4N+1}) = \text{lcs}^\rho(r_X L_X) \) and \( \text{lcs}^\rho(r_X L_X^{\leq 4N+1}) = \text{lcs}^\rho(r_X L_X) \). Thus, \( T_X^{\leq 4N+1} \approx_{\text{ia}} T_{\text{ias}}(r_X L_X^{\leq 4N+1}) \approx_{\text{ia}} T_{\text{ia}}(r_X L_X) \).

Assume thus that \( G \) is not \( \text{wf} \).

We assume that all nullary rules \( X \rightarrow \pi v \) are already reduced and w.l.o.g. \( \text{wwf} \).

Further w.l.o.g. \( G \) is nonnegative.

Then there is some \( \alpha \in L(G) \) that is not \( \text{wf} \).

As \( r_S = \epsilon \) and \( G \) is nonnegative, we cannot have

We show that there is some rule \( X \rightarrow G Y Z \) and words \( \alpha_X = \alpha_Y \alpha_Z \) with \( \alpha_Y \in L_Y \) and \( \alpha_Z \in L_Z \) s.t.:

- \( r_X \alpha_Y \text{r}_Z \) is not \( \text{wf} \)
- \( r_X \text{r}_Z \) is \( \text{wf} \)
- \( r_Y \alpha_Y \) is \( \text{wf} \)

To this end, consider any derivation of \( \alpha \):
Balancedness

Set $X := S$ and $\alpha_X := \alpha$

We have $r_X = r_S = \varepsilon$ with $r_X \alpha_X$ not wf.

While $r_X \alpha_X$ is not wf:

Then there some rule $X \rightarrow_G Y Z$ and factorization $\alpha_X = \alpha_Y \alpha_Z$ as by assumption $r_X r$ is wf for all constant rules $X \rightarrow_G r$.

If $r_Y \alpha_Y$ is not wf:

Redefine $X := Y$ and $\alpha_X := \alpha_Y$ and descend accordingly into the derivation tree of $\alpha_Y$.

If $r_Z \alpha_Z$ is not wf:

Redefine $X := Z$ and $\alpha_X := \alpha_Z$ and descend accordingly into the derivation tree of $\alpha_Z$.

Otherwise $r_Z \alpha_Z$ is wf, thus $\alpha_Z \cong \pi_Z \nu_Z$ with $r_Z = r'_Z \nu_Z$.

Thus $r_X \alpha_Y r_Z$ is not wf as $r_X \alpha_X = r_X \alpha_Y \alpha_Z \cong r_X \alpha_Y r_Z r'_Z \nu_Y$.

So, there is some least derivation height $n_0$ s.t.

- $r_X L_X^{< n_0}$ is wf for every nonterminal $X$
- $r_X \gamma_Y$ is wf for all rules $X \rightarrow_G Y Z$
- there exists a nonterminal $X_0$ with $X_0 \rightarrow_G Y Z$, $\alpha_Y \in L_Y^{< n_0}$, and $r_X, \alpha_Y \gamma_Z$ not wf anymore.

As all $r_Y L_Y^{< n_0}$ are wf, we have $\rho(r_Y L_Y^{< n_0}) \approx_{lcs} T_{\alpha}(r_Y L_Y^{< n_0})$.

Thus also $\rho(r_X L_X^{< n_0}) = \rho(r_X \gamma_Y) \rho(r_Y L_Y^{< n_0}) \approx_{lcs} \rho(r_X \gamma_Y) T_{\gamma_Y}^{< n_0}$.

Finally, as $G$ is nonnegative, also $r_X L_X^{< n_0} \gamma_Z$ is nonnegative.

Thus, as $r_X \alpha_Y \gamma_Z$ is not wf, we have that $lcs'((r_X L_X^{< n_0}) \gamma_Z)$ is not wf, and thus $\rho(r_X \gamma_Y) T_{\gamma_Y}^{< n_0} \gamma_Z$ is not wf.

So, if $n_0 \leq 4N + 1$, by iteratively computing $T_X^{< h} \approx_{lcs} T_{\alpha}(\rho(r_X L_X^{< h})))$, we discover the error.

Otherwise $r_Z \not\subseteq lcs'((r_X L_X^{< 4N+1}) \gamma_Z)$ but $r_Z \not\subseteq lcs'((r_X L_X^{< n_0}) \gamma_Z)$.

Thus $\rho(r_X L_X^{< 4N+1}) \not\approx_{lcs} r_X L_X^{< n_0}$ and thus $r_Y L_Y^{< 4N+1} \not\approx_{lcs} r_Y L_Y^{< n_0}$.

So, for at least one nonterminal lcssum cannot have converged.

$\blacksquare$