Computing the longest common prefix of a context-free language in polynomial time

Michael Luttenberger, Raphaela Palenta, and Helmut Seidl
TU München, Germany, {luttenbe,palenta,seidl}@in.tum.de

February 23, 2017

We present two structural results concerning longest common prefixes of non-empty languages. First, we show that the longest common prefix of the language generated by a context-free grammar of size $N$ equals the longest common prefix of the same grammar where the heights of the derivation trees are bounded by $4N$. Second, we show that each non-empty language $L$ has a representative subset of at most three elements which behaves like $L$ w.r.t. the longest common prefix as well as w.r.t. longest common prefixes of $L$ after unions or concatenations with arbitrary other languages. From that, we conclude that the longest common prefix, and thus the longest common suffix, of a context-free language can be computed in polynomial time.

1 Introduction

Let $\Sigma$ denote an alphabet. On the set $\Sigma^*$ of all words over $\Sigma$, the prefix relation provides us with a partial ordering $\sqsubseteq$ defined by $u \sqsubseteq v$ iff $uu' = v$ for some $u' \in \Sigma^*$. The longest common prefix (lcp for short) of a non-empty set $L \subseteq \Sigma^*$ then is given by the greatest lower bound $\bigsqcap L$ of $L$ w.r.t. this ordering. For two words $u, v \in \Sigma^*$, we also denote this greatest lower bound as $u \sqcap v$. Our goal is to compute the lcp when the language $L$ is context-free, i.e., generated by a context-free grammar (CFG).

The computation of the lcp arises naturally in various applications of context-free languages. As a first example, recall that the possible (unevaluated) runs of a recursive program can be described by a context-free grammar: the nonterminals of the grammar are then in one-to-one correspondence to the nodes of the interprocedural control-flow

\footnote{It is folklore how to efficiently compute from a given CFG for $L$ an CFG for its reversal $L^R = \{a_1a_{-1} \ldots a_2a_1 \mid a_1a_2 \ldots a_{-1}a_1 \in L\}$; hence, computing the longest common suffix of a context-free language is trivially reducible in polynomial time to the computation of the lcp of a context-free language.}
graph, and the lcp of the language generated by any nonterminal is the longest common sequence of statements always executed whenever the respective node in the control-flow graph is reached. CFGs also represent a popular formalism to specify which words are well-formed and which can be rejected immediately. Assume that we are given a CFG for the legal outputs of a program. This CFG might be derived from the specification as well as from an abstract interpretation of the program. Then the lcp of this language represents a prefix which can be output already, before the program actually has been run. This kind of information is crucial for the construction of normal forms, e.g., of string producing processors such as linear tree-to-string transducers [1, 6]. For these devices, the normal forms have further interesting applications as they allow for simple algorithms to decide equivalence [2] and enable efficient learning [7].

Obviously, the lcp of the context-free language $L$ is a prefix of the shortest word in $L$. Since the shortest word of a context-free language can be effectively computed, the lcp of $L$ is also effectively computable. The shortest word generated from a context-free grammar $G$, however, may be of length exponential in the size of $G$. Therefore, it is an intriguing question whether or not the lcp can be efficiently computed.

Here, we show that the longest common prefix can in fact be computed in polynomial time. As the words the algorithm computes with, may be of exponential length, we have to resort to compressed representations of long words by means of straight-line programs (SLPs) [9]. We will rely on algorithms for basic computational problems for straight-line programs as presented, e.g., in [8].

Our method of computing $\bigcap L$, is based on two structural results. First we show in Section 3 that it suffices to consider the finite sublanguage of $L$ consisting of those words, for which there is a derivation tree of height at most $4N$ — with $N$ the number of nonterminals of the CFG for $L$. This implies that (1) for the proofs we can replace the grammar by an acyclic context-free grammar, and (2) the actual fixpoint iteration will converge within at most $4N$ iterations. Second we show in Section 4 that for every non-empty language $L$ there is a subset $L' \subseteq L$ of at most three elements which is equivalent to $L$ w.r.t. the lcp after arbitrary concatenations with other words. This means that for every word $w$, the language $L'w$ has the same lcp as $Lw$.

We illustrate both results by examples. For the first result, i.e. the restriction to derivation trees of bounded height, consider the language

$$L := \{a^2 b (a^2 b)^i a^2 b (a^2 ba)^i a^2 ba^2 ba^3 \mid i \in \mathbb{N}_0\}$$

generated by the context-free grammar consisting of the following rules over the alphabet $\Sigma = \{a, b, c\}$ and the six nonterminals $\{S, X, A_2, A_1, X_2, X_1\}$:

$$S \rightarrow X_2 A_2 b A_2 b A_2 a \quad A_2 \rightarrow a A_1 \quad A_1 \rightarrow a \quad X \rightarrow A_2 b$$
$$X_2 \rightarrow a X_1 \quad X_1 \rightarrow a b X \quad X \rightarrow X_2 A_2 b a$$

It is easy to check that here the lcp is already determined by repeating the derivation of $X$ to $aabXaaba$ at most two times which correspondence to the sublanguage consisting
of all words which have a derivation tree of height at most 9.

\[
\begin{align*}
\bigcap L &= aabaabaabaaba a \quad (i = 0) \\
\bigcap aabaabaabaaba aabaabaa & \quad (i = 1) \\
\bigcap aabaabaabaaba baaaabaabaaba & \quad (i = 2) \\
\bigcap aabaabaabaaba baaaabaabaabaaba & \quad (i = 3) \\
\bigcap aabaabaabaaba b \ldots & \quad (i \geq 4) \\
\end{align*}
\]

\[
= aabaabaabaaba
\]

We remark that the bound of \(4N\), i.e. 24 for this example, on the height resp. the number of iterations needed to converge is a crude overapproximation based on the pigeon-hole principle which does not take into account the structure of the grammar. The actual computation of the lcp may thus terminate much earlier. We discuss this in more detail in Example 3.

We show in Section 4 that for every language \(L\) there is a unique longest, possibly infinite, word \(w_L\) satisfying \(\bigcap (Lw) = (\bigcap L)w\). Based on this observation, every language can be equivalently represented by a sublanguage consisting of at most three words where these three words have to be chosen such that both the lcp and \(w_L\) are preserved.

We can choose \(L'_3 = \{aba, abab, ababa\}\) as an equivalent finite sublanguage; note that \(L'_3 = \{aba, abab\}\) is not equivalent to \(L_3\) as it is quasi-periodic with \(w_{L'_3} = b^2\).

In order to compute the lcp of a given context-free language \(L\) we then (implicitly) unfold the given context-free grammar into an acyclic grammar, and compute for every nonterminal of the unfolded grammar an equivalent sublanguage of at most three words, each compressed by means of a SLP, instead of the actual language. From this finite representation of \(L\) we then can easily obtain its lcp. Altogether, we arrive at a polynomial time algorithm.

2 Preliminaries

\(\Sigma\) denotes a (finite) alphabet. \(\Sigma^*\) is the set of all finite words over \(\Sigma\) with \(\varepsilon\) the empty word, \(\Sigma^\omega\) the set of all (countably) infinite words over \(\Sigma\). We use \((\omega-)\)rational expressions to denote words and languages, e.g. \(w^* = \varepsilon + w + w^2 + \ldots = \sum_{i \in \mathbb{N}_0} w^i\) and \(w^\omega = \underbrace{w_0w_1w_2w_3\ldots}_{\infty}\).

By \(C_\Sigma = \{(u, v) \in \Sigma^* \times \Sigma^*\}\) we denote the set of all pairs of finite words over \(\Sigma\). We define a multiplication on \(C_\Sigma\) by \((x, x')(y, y') := (xy, yx')\). For \((x, x') \in C_\Sigma\) and \(w \in \Sigma^*\) set \((x, x')w = xw\overline{x}\).
For $u, v \in \Sigma^*$ we write $u \subseteq v$ ($u \varsubsetneq v$) to denote that $u$ is a (strict) prefix of $v$, i.e. $v = uw$ for some $w \in \Sigma^*$ ($w \in \Sigma^+$). For $L \subseteq \Sigma^*$ (with $L \neq \emptyset$) its longest common prefix $\text{lcp} \prod L$ is given by the greatest lower bound of $L$ w.r.t. this ordering. We simply write $u \sqcap v$ for $\sqcap \{u, v\}$. Note that for any word $w \in L$ there is at least one word $\alpha \in L$ s.t. $\prod L = w \sqcap \alpha$; we call any such $\alpha$ a witness (w.r.t. $w$). Note that $\sqcap$ is commutative and associative; concatenation distributes from the left over the $\text{lcp}$ (i.e. $u(v \sqcap w) = uv \sqcap uw$); and the $\text{lcp}$ is monotonically decreasing on the union of languages, i.e. $\sqcap (\prod (L \cup L')) = (\sqcap \prod L) \sqcap (\sqcap \prod L')$. The $\text{lcp}$ of infinite words is defined analogously.

A word $p \in \Sigma^*$ is called a power of a word $q$ if $p = q^*$; then $q$ is called a root of $p$; if $p \neq \varepsilon$ is its own shortest root, $p$ it is called primitive. We recall two well-known results:

**Lemma 1** (Commutative Words, [3]). Let $u, v \in \Sigma^*$ be two words. If $uv = vu$, then $u, v \in p^*$ for some primitive $p \in \Sigma^*$.

**Lemma 2** (Periodicity lemma of Fine and Wilf, [5]). Let $u, v \in \Sigma^+$ be two non-empty words. If $|uw \sqcap v^\omega| \geq |u| + |v| - \gcd(|u|, |v|)$, then $uv = vu$.

Combining these two lemmata yields the following result which is an useful tool in the proofs to follow:

**Corollary 1.** Let $u, v \in \Sigma^*$ with $uv \neq vu$.

Then $w^\omega \sqcap v^\omega = uv \sqcap vu$ with $|uv \sqcap vu| < |u| + |v| - \gcd(|u|, |v|)$.

**Proof.** Since the bound of the size of $|uv \sqcap vu|$ follows from Lemma 2 we only have to show that $uv \sqcap vu = u^\omega \sqcap v^\omega$. If $|u| = |v|$, then $uv \neq vu$ implies $u \neq v$ and $uv \sqcap vu = u \sqcap v = u^\omega \sqcap v^\omega$.

W.l.o.g. we assume that $|u| < |v|$. As $uv \neq vu$, we have $\varepsilon \neq u$. Let $v \sqcap u^\omega = u^k u' \sqcap v^{k+1}$ with $v = u^k u' v'$ and $u = u' u''$. It follows that $uv \sqcap vu = uu^k u' v' u \sqcap uu^k u' v' u = uu^k (uu' v' \sqcap u' v' \sqcap v' u' u'')$.

If $v' \neq \varepsilon$, we have $u'' u' v' \sqcap v' u' u'' = u'' \sqcap v' = \varepsilon$, and thus $uv \sqcap vu = u^k u' = u \sqcap v = v^\omega \sqcap u^\omega$.

So assume $v' = \varepsilon$, i.e. $v \sqcap u^\omega$ with $k > 0$ as $|u| < |v|$. As $uv = u^k u' u'' u' \neq u^k u' u'' u' = vu$, also $u' u'' \neq u' u''$. Hence $uv \sqcap vu = u^k u' (u' u'' \sqcap u' u'') = u^{k+1} u \sqcap vu = u^\omega \sqcap v^\omega$, which concludes the proof. \hfill $\square$

Here is a short example for the last corollary:

**Example 1.** Let $u = aab$, $v = aaba = ua$. Then $uv \sqcap vu = aaba \sqcap aabaab = aabaa = va$ and $u^\omega \sqcap v^\omega = aabaababv^\omega \sqcap aabaabav^\omega = aaba$ with $|aaba| = |u| + |v| - \gcd(|u|, |v|) - 1$. I.e. the bound is sharp. Note that this example also shows, that even if $uv \neq vu$ and $\varepsilon \neq u \sqsubseteq v$, we still can have $v \sqsubseteq uv$.

We briefly discuss properties of the $\text{lcp}$ for very simple regular languages. These will be used several times in the proofs of Section 4 in order to bound the height of the derivation trees we need to consider:

**Lemma 3.** Let $y \neq \varepsilon$, then $w \sqcap yw = \sqcap y^* w = w \sqcap y^\omega$. 

\hfill 4
We write Example 2. Let \( w \cap y^\omega = y^k y' \sqsubseteq y^{k+1} \) with \( w = y^k y' \). Then for any \( i > 0 \) we have \( w \cap y^i w = w \cap y^k y^{i+1} y' = y^k y' = w \cap y^{k+1} = w \cap y^\omega \). \( \square \)

Lemma 4. If \( w \not\subseteq yw \), then \( \bigcap y^w w = w \cap yw \subseteq w \).

Proof. Since \( w \not\subseteq yw \), \( w \neq \varepsilon \) and \( y \neq \varepsilon \). Assume that \( w = w \cap yw \), then by the preceding lemma \( w = w \cap yw = w \cap y^\omega \), i.e. \( w \cap y^\omega \) and thus \( w \subseteq yw \). \( \square \)

3 LCP of a context-free language

Our main result in this section, Theorem 2, shows that for every context-free language \( L = L(G) \) generated by the given CFG \( G \) its \( \text{lcp} \) \( \bigcap L \) is equal to the \( \text{lcp} \) of its finite sublanguage \( L' \) which contains only those words \( w \in L \) which possess a derivation tree w.r.t. \( G \) whose height (considering only nonterminals) is at most four times the number of nonterminals of \( G \). For the main result we require the following, very technical lemma.

Theorem 1. Let \( L = \langle x, x \rangle \rangle \{(y_1, y_1) + \cdots + (y_l, y_l) \}^* \) for \( \langle x, x \rangle \rangle \{y_1, y_1, \ldots, (y_l, y_l) \} \in \mathcal{C}_{\Sigma} \) and \( w \in \Sigma^* \).

Then: \( \bigcap L = (x, x) \rangle \rangle (y_1, y_1)^{\leq 2} + \cdots + (y_k, y_k)^{\leq 2} \rangle \rangle w \)

Further, if \( \bigcap L = xw\rangle \rangle xyw^2y^2x \subseteq xw\rangle \rangle xyw^k \) for some \( (y, y) \in \{y_1, y_1, \ldots, (y_l, y_l) \} \), then w.r.t. this \( y \) there exists some primitive \( q \in \Sigma^* \) and some \( k > 0 \) such that \( yw = wq^k \land q^l \not\subseteq yq \land \bigcap L = xw\rangle \rangle xyw^kq^k \wedge xwq^k(y \rangle \rangle q^\omega) \subseteq \bigcap L \wedge xwq^k+1(y \rangle \rangle q^\omega) \)

The proof of the main theorem of this section, Theorem 2, crucially depends on the observation that in the case \( \bigcap L \subseteq xw\rangle \rangle xyw^\infty \), all the words \( y_i \) are powers of the same primitive word \( p \) with \( pw = wq \) and all that is needed to obtain a witness is one additional power of \( p \) resp. its conjugate \( q \) (with \( pw = wq \)) to which Theorem 1 refers to. We give an example in order to clarify the statement of Theorem 1 in the case of \( l = 2 \wedge (y_1, y_2) = (y_2, y_1) \) which is central to Theorem 2.

Example 2. We write \( (y, y) \) for \( (y_1, y_1) \) and \( (z, z) \) for \( (y_2, y_2) \), respectively. Let \( (x, x) = (\varepsilon, ababa) = (\varepsilon, qqqaa) = (\varepsilon, \varepsilon) = (ab, ababab) = (q, quaab), (z, z) = (ab, ababab) = (q, qbaab), (w, w) = \varepsilon \) with \( q = ab = y = z \). We then have:

\[
\begin{align*}
wx &= \varepsilon, & xy &= \varepsilon, & xw &= ababa, & wy &= ababa, & zw &= ababa.
\end{align*}
\]

\[
\begin{align*}
xw &= ababa, & xyy &= ababaabababa, & xzw &= ababaabababa, & xyy &= ababaabababa, & xzw &= ababaabababa.
\end{align*}
\]

\[
\begin{align*}
xw &= ababaabababa, & xyy &= ababaabababa, & xzw &= ababaabababa, & xyy &= ababaabababa, & xzw &= ababaabababa.
\end{align*}
\]

\[
\begin{align*}
x+y+z &= \varepsilon, & xyz &= ababab, & xw &= ababaabababa, & xw &= ababaabababa.
\end{align*}
\]

\[
\begin{align*}
x &= \varepsilon, & xy &= \varepsilon, & xw &= ababa, & xz &= ababaabababa.
\end{align*}
\]

\[
\begin{align*}
x &= ababaabababa, & xz &= ababaabababa.
\end{align*}
\]
So in this example, any word except for \( xyw\tilde{y}\tilde{e} \) and \( xzw\tilde{z}\tilde{e} \) is a witness for the \( \text{lcp} \) w.r.t. \( xw\tilde{x} \). W.r.t. the proof of Theorem 3 it is important that also in general we can pick a witness which either is derived using only \((y, \tilde{y})\) or \((z, \tilde{z})\) but not both, and that we need to use \((y, \tilde{y})\) resp. \((z, \tilde{z})\) at most twice in order to get one additional copy of the conjugate \( q \) of the primitive root of both \( y \) and \( z \).

To give an impression of the proof of Theorem 4 we show the case \( l = 1 \). The complete proof of Theorem 4 can be found in the appendix.

**Lemma 5.** Let \( L = (x, \tilde{x})(y, \tilde{y})^*w \). Then: \( \bigcap L = \bigcap (x, \tilde{x})(y, \tilde{y})^2w \).

If \( \bigcap L \subseteq xw\tilde{x} \cap xyw\tilde{y}x \), then there is some primitive \( q \) and some \( k > 0 \) s.t.

\[
yw = \bar{w}^k \land q\tilde{y} \neq \bar{y}q \land \bigcap L = xw\tilde{x} \cap xyw\tilde{y}x \land xwq^k(\bar{y} \cap q^\omega) \subseteq \bigcap L \subseteq xwq^{k+1}(\bar{y} \cap q^\omega)
\]

**Proof.** If \( \epsilon \lor \bar{y} = \epsilon \) then by Lemma 4 \( \bigcap L = \bigcap xwy^*\tilde{x} = \bigwedge (\tilde{x} \cap \bar{y}x) = xw\tilde{x} \cap xyw\tilde{y}x \), respectively \( \bigcap L = \bigcap xy^*w\tilde{x} = \bigwedge (w\tilde{x} \cap yw\tilde{y}x) = xw\tilde{x} \cap xyw\tilde{y}x \). Hence, assume that \( y \neq \epsilon \) and \( \bar{y} \neq \epsilon \) in the following.

If \( w \not\subseteq yw \), then from Lemma 4 we have \( w \cap y^*w = \bigcap yw = \bigcap y^\omega \cap w \), and thus

\[
\bigcap L = \bigcap xwy^k\tilde{y}x = \bigwedge y^*w = \bigwedge (w \cap yw) = xw\tilde{x} \cap xyw\tilde{y}x.
\]

So assume that \( w \subseteq yw \) from now on. Then there is some \( \tilde{y} \) s.t. \( w\tilde{y} = yw \). Let \( q \) be the primitive root of \( y \) s.t. \( \bar{y}q = w\tilde{y} \). Thus for \( i \) large enough that \( |w\tilde{x}| \leq i |y| \), we have

\[
\bigcap L \subseteq xw\tilde{x} \cap xy^i\tilde{y}w^i\tilde{x} = xw\tilde{x} \cap xwq^k\tilde{y}^i\tilde{x} = xw(\tilde{x} \cap q^\omega) \subseteq xwq^\omega.
\]

We factorize \( \tilde{x} \) and \( \tilde{y} \) w.r.t. \( q^\omega \):

- Let \( \tilde{x} = q^0q'\tilde{x}' \) with \( \tilde{x} \cap q^\omega = q^0q' \cap q^{n+1} \).
- Let \( \tilde{y} = q^k\tilde{y}'q' \) with \( \tilde{y} \cap q^\omega = q^k\tilde{y}' \cap q^{k'+1} \).

If \( q^0q' \subseteq q^{k+k'} \tilde{q} \), then

\[
\bigcap L = xw\tilde{x} \cap xyw\tilde{y}x = xw(q^0q'\tilde{x}' \cap q^{k+k'}\tilde{y}'\tilde{x}) = xw(q^0q'\tilde{x}' \cap q^{k+k'}q^\omega) = xw(q^\omega q'x' \cap q^{k+k'}q^\omega) = xw(q^\omega q'x') = xw(\tilde{x} \cap q^\omega)
\]

Thus \( q^k(\tilde{y} \cap q^\omega)\tilde{x} \subseteq q^\omega \tilde{q} = \tilde{x} \cap q^\omega \) for the following.

If \( q\tilde{y} = \tilde{y}q \), then \( \tilde{y} = q^k\tilde{y} \), and \( xyw\tilde{y}x \) is a witness as \( k > 0 \)

\[
\bigwedge xw\tilde{x} \cap xy^i\tilde{y}w^i\tilde{x} = xw(q^0q'\tilde{x}' \cap q^{k+k'}q^\omega q^\omega) = xw(q^0q'\tilde{x}' \cap q^{n+1}) = xw\tilde{x} \cap xyw\tilde{y}x
\]

We therefore also assume \( q\tilde{y} \neq \tilde{y} \) from here on.

If \( \tilde{x} \cap q^\omega = q^0q' = q^{k+k'}\tilde{q} = q^k(\tilde{y} \cap q^\omega) \), then \( xyyw\tilde{y}x \) is a witness:

\[
xw\tilde{x} \cap xyw\tilde{y}x = xw(q^0q'\tilde{x}' \cap q^{2k+k'}q^\omega) = xwq^\omega(q^0q'x') = xwq^\omega(\tilde{y} \cap q^\omega) = xwq^\omega(\tilde{y} \cap q^\omega) \subseteq xwq^{k+1}(\tilde{y} \cap q^\omega)
\]
Note that we only need at most one additional $q$ (coming from $y = q^k$), i.e. also $\bigcap L = xw\bar{x} \cap xywq\bar{y}x$.

It remains the case $q^{k+k'}\hat{q} \sqsubseteq q^kq'$, i.e. $q^k\bar{y} \cap q^\omega \sqsubseteq \bar{x} \cap q^\omega$. Hence $0 < k \leq k + k' \leq n$. Define $\phi \sqsubseteq q^\omega$ by $\bigcap L = xw\phi$. From $q^{k+k'}\hat{q} \sqsubseteq q^\omega q'$ it follows that $xwq^{k+k'}\hat{q} \sqsubseteq xw\bar{x} \cap xyiwyj\bar{x}$ for all $i \in \mathbb{N}_0$, implying $\bar{y} \cap q^\omega = q^{k+k'}\hat{q} \sqsubseteq \phi$.

If $\phi = q^{k+k'}\hat{q}$, then we have $\bigcap L = xw\bar{x} \cap xyw\bar{y}x$: from $xwq^{2k+k'}\hat{q} \sqsubseteq xyiwyj\bar{x}$ for $i > 1$ and $q^{k+k'}\hat{q} \sqsubseteq q^\omega q'$ it follows that $xw\bar{x} \cap xyiwyj\bar{x}$ is a strictly longer prefix of $xwq^\omega$ than $xwq^{k+k'}\hat{q}$ as $k > 0$.

So assume $q^{k+k'}\hat{q} \sqsubseteq \phi$. If $\bar{y}' \neq \varepsilon$, then $xw\bar{x} \cap xyw\bar{y}x = xwq^{k+k'}\hat{q}$ as $q^{k+k'}\hat{q} \sqsubseteq q^\omega q'$. Thus also $\bar{y}' = \varepsilon$ resp. $\bar{y} = q^{k'}\hat{q}$ for the remaining.

As $q\bar{y} = q^{k'}\hat{q}\bar{q} \neq q^{k'}\hat{q}q$ (recall $q = \hat{q}\hat{q}$) also $q\bar{q} = \hat{q}q$ and thus

\[
xwq^{k+k'}\hat{q} \sqsubseteq xw\phi \sqsubseteq xwq^{k+k'}\hat{q} \sqsubseteq xwq^{k+k'}(\hat{q}q^nq'\bar{x} \cap q^k\hat{q}\bar{y}\bar{x}) \forall n \geq 0
\]

Hence $q^{k+k'}\hat{q} \sqsubseteq \phi \sqsubseteq q^{k+k'+1} \hat{q}$. That is either $xyw\bar{y}x$ or $xyyw\bar{y}x$ has to be a witness as we can extend $q^{k+k'}\hat{q}$ by at most $|q| - 1$ symbols, i.e. only by a strict prefix of $\hat{q}\bar{q}$; but as $k > 0$, this additional $q$ is again given by $xyyw\bar{y}x$. In particular, we have again that, if $xyyw\bar{y}x$ is a witness, then so is $xywq\bar{y}x$, i.e. we only require one additional power of $q$ left of $\bar{y}$ resp. $\hat{q}$.

\[\square\]

Using Theorem 1, we show:

**Theorem 2.** Let $L = L(G)$ be given by a context-free grammar $G = (\Sigma, V, P, S)$ in Chomsky normal form. Let $\mathcal{L} \subseteq L$ be the finite language of all words of $L$ for which there is a derivation tree w.r.t. $G$ of height\(^2\) at most $4|V|$ with $V$ the nonterminals of $G$. Then: $\bigcap L = \bigcap \mathcal{L}$.

**Proof.** Let $L = L(G)$ with $G$ CFG in CNF. Let $N$ be the number of nonterminals of $G$. Let $\sigma \in L$ be a shortest word, and $\alpha \in L$ a shortest word with $\bigcap L = \sigma \cap \alpha$. Set $\pi := \bigcap L$.

We claim that there is at least one such $\alpha$ (for any fixed $\sigma$) that has an derivation tree w.r.t. $G$ of height less than $4N$ (not counting the leaves representing the letters of a word). If $\sigma = \alpha$, we are done as $\sigma$ has a derivation tree of height less than $N$. So assume $\sigma \neq \alpha$ s.t. $\sigma = \pi\alpha\sigma'$ and $\alpha = \pi\beta\alpha'$ with $a \neq b$ alphabet symbols, and fix any derivation tree $t$ of $\alpha$ w.r.t. $G$.

We will show the stronger claim, namely that any path from the root of $t$ to any letter of $\pi b$ has length at most $3N$ (i.e. all the paths leading to the separating letter $b$ or a letter left of it, see Figure 1); note that any path that leads to a letter right of $b$ (i.e. into $\alpha'$) has to enter a subtree of height less than $N$ as soon as it leaves the path leading to $b$ because of the minimality of $\alpha$. Hence, if all the paths leading to $b$ or a letter left of $b$ have length less than $3N$, the longest path in the derivation tree must have length at most $4N$.

\(^2\)We measure the height of a derivation tree only w.r.t. nonterminals along a path from the root to a leaf.
Figure 1: Factorization of a witness $\alpha = (x, \bar{x})(y_1, \bar{y}_1)(y_2, \bar{y}_2)(y_3, \bar{y}_3)w = \pi b a'$ w.r.t. a nonterminal $A$ occurring at least four times along a path (here the dashed path) in a derivation tree of $\alpha$ leading to a letter either within the LCP $\pi = \prod L$ or to the letter, here $b$ (the leaf of the dotted path), that bounds the LCP.

So assume there is a path leading to a letter within $\pi b$ that has at least length $3N$ i.e. consists of at least $3N + 1$ nonterminals. Then there is one nonterminal $A$ that occurs at least four times leading to a factorization $\alpha = (x, \bar{x})(y_1, \bar{y}_1)(y_2, \bar{y}_2)(y_3, \bar{y}_3)w$

Note that $x \bar{x} \neq \varepsilon$, $y_i \bar{y}_i \neq \varepsilon$ ($i = 1, 2, 3$), and $w \neq \varepsilon$ as $G$ is in CNF. As this path ends at $b$ or left of it, we have $xy_1y_2y_3 \subseteq \pi$ implying that $y_i y_j = y_j y_i$ for all $i, j \in \{1, 2, 3\}$, so $y_i = p^{k_i}$ for the same primitive $p$.

Let $L' = (x, \bar{x})[(y_1, \bar{y}_1) + (y_2, \bar{y}_2) + (y_3, \bar{y}_3)]^* w$. By construction $L' \subseteq L$ and thus $\bigcap L = \bigcap L' \subseteq xw\bar{x} \cap \alpha$. As $xw\bar{x}$ is shorter than $\alpha$, it cannot be a witness, so $\pi a \subseteq xw\bar{x}$ and $\pi = xw\bar{x} \cap \alpha$. Hence

$$\bigcap L = \sigma \cap \alpha = xw\bar{x} \cap \alpha = \bigcap L'$$

It therefore suffices to consider $L'$ in the following; in particular, $\alpha$ has to be a witness w.r.t. $xw\bar{x}$ of minimal length, too.

By virtue of Theorem 1

$$\bigcap L' = \bigcap (x, \bar{x})[(y_1, \bar{y}_1) \leq 2 + (y_2, \bar{y}_2) \leq 2 + (y_3, \bar{y}_3) \leq 2] w$$

Note that for any $i = 1, 2, 3$

$$\bigcap L' \subseteq xw\bar{x} \cap x y_i w_{y_i} \bar{x}$$

as $|x y_i w_{y_i} \bar{x}| < |\alpha|$ and thus $x y_i w_{y_i} \bar{x}$ cannot be a witness by minimality of $\alpha$. So for some $I \in \{1, 2, 3\}$

$$\bigcap L' = xw\bar{x} \cap x y_I y_I w_{y_I} \bar{y}_I \bar{x} \subseteq \alpha$$

i.e. $x y_I y_I w_{y_I} \bar{y}_I \bar{x}$ has to be also a witness.
Set \((y, \tilde{y}) := (y_I, \tilde{y}_I)\) and
\[ L'' = (x, \bar{x})(y, \tilde{y})^* w \]
so that \(L'' \subseteq L' \subseteq L\) and \(\bigcap L = \bigcap L' = \bigcap L''\) as
\[ xw\bar{x} \cap xyyw\tilde{y}\tilde{y}x = \bigcap L \subseteq \bigcap L' \subseteq \bigcap L'' \subseteq xw\bar{x} \cap xyyw\tilde{y}\tilde{y}x \subseteq xyw\tilde{y}x \]
Applying Theorem 1 to \(yw = wq_k \land q\tilde{y} \neq \tilde{y}q\land \bigcap L = \bigcap L'' = xw\bar{x} \cap xywq\tilde{y}\tilde{y}x \subseteq \bigcap L \subseteq xwq^{k+1}(\tilde{y} \cap q'\cap \tilde{y} q') \]
for the primitive \(q\) satisfying \(pw = wq\). As \(y_i = p^k_i\) for all \(i \in \{1, 2, 3\}\), also \(y_i w = wq^k_i\).

As \(xwq^k \subseteq \bigcap L \subseteq xwq^{\epsilon}\), we find some \(m \geq 0\) and \(q' \subseteq q\) s.t. \(\pi = \bigcap L = xwq^k q^m \tilde{q}\). As \(xyw\tilde{y}x\) is not a witness, but \(\bigcap L = xw\bar{x} \cap xyyw\tilde{y}\tilde{y}x = xw\bar{x} \cap xyw\tilde{y}x\):
\[
\begin{align*}
\pi a &= xwq^k q^m a = xw\bar{x} \\
\pi a &= xwq^k q^m a = xyw\tilde{y}x = xwq^k \tilde{y} x \\
\pi b &= xwq^k q^m \tilde{q} = xyw\tilde{y}x \\
\pi b &= xwq^k q^m \tilde{q} = xwq^{k+1} \tilde{y} \tilde{x}
\end{align*}
\]
Hence \(\tilde{q} \subseteq q\) as \(q^m a \subseteq \tilde{y} x\) and \(q^m \tilde{q} \subseteq q\tilde{y} \subseteq qq^m \tilde{q} a\). Using \(\bigcap L \subseteq xwq^{k+1}(\tilde{y} \cap q'\cap \tilde{y} q')\), we further obtain \(q^m \tilde{q} \subseteq q(\tilde{y} \cap q')\).

If there was at least one \(j \in \{1, 2, 3\} \setminus \{I\}\) with \(k_j > 0\) s.t. \(y_j = p^k_j \neq \epsilon\), then \((x, \bar{x})(y_j, \tilde{y}_j)(y, \tilde{y})w\) would be a witness shorter than \(\alpha\) as:
\[
(x, \bar{x})(y_j, \tilde{y}_j)(y, \tilde{y})w = x y_j w y_w y_j \tilde{x} \supseteq x w q^{k_j \tilde{y}}(\tilde{y} \cap q'\cap \tilde{y} q') = q^m \tilde{q} \tilde{b}
\]
So for all remaining \(j \in \{1, 2, 3\} \setminus \{I\}\) we have \(y_j = \epsilon\) and thus \(\tilde{y}_j\).

As \(\pi a = xwq^{k+m} q a \subseteq x y_j w y_j \bar{x} = x w y_j \bar{x}\), it follows that \(q^{k+m} q a \subseteq \tilde{y}_j \bar{x} = \tilde{y}_j q^{k+m} q a \bar{x}'\).

By Lemma 3 \(x w y_j^2 \bar{x} = x w \bar{x} \cap x w y_j \tilde{x}\), hence \(\pi a \subseteq x w y_j^2 \bar{x}\), i.e. \(q^{k+m} q a \subseteq \tilde{y}_j \bar{a}\).

If \(q^m \tilde{q} \subseteq \tilde{y}_j\) for some \(j \in \{1, 2, 3\} \setminus \{I\}\) (recall \(\tilde{q} \subseteq q\), then as \(a \neq b\)
\[
x w \bar{x} \cap (x, \bar{x})(y, \tilde{y})(y_j, \tilde{y}_j)w = x w (x \cap q^k \tilde{y} \tilde{y} \tilde{x}) = x w q^{k+m} q a \cap q^{k+m} \tilde{q} b = \pi
\]
i.e. \(x y y_j w y_j \tilde{y} \tilde{x}\) would be a shorter witness than \(\alpha\). Hence \(\tilde{y}_j \subseteq q^m \tilde{q} \subseteq q^{k+m} q a\) for both \(j \in \{1, 2, 3\} \setminus \{I\}\). Thus:
\[
|q^e \cap \tilde{y}_j| \geq |q^{k+m} \tilde{q}| \geq |q| + |q^m \tilde{q}| > |q| + |\tilde{y}_j| - \gcd(|q|, |\tilde{y}_j|)
\]
By the periodicity lemma of Fine and Wilf (Lemma 2) this implies \(\tilde{y}_j = q^{k_j'}\) for some \(k_j' > 0\) (as \(q\) primitive), and, subsequently as the final contradiction, that \(x y y_j \bar{w}_j \tilde{y}_j \bar{x}\) would be a shorter witness. \(\square\)
4 Small Equivalent Subsets of Languages

In this section we formally introduce a notion of equivalence of languages w.r.t. longest common prefixes. The first main result of this section is that every non-empty language has an equivalent subset consisting of at most three elements. In case of acyclic context-free languages, such a subset can be computed in polynomial time. In combination with Theorem 2 we can lift the restriction on acyclicity. This enables us to ultimately conclude that the longest common prefix of a context-free language can be computed in polynomial time.

Definition 1. We say that a language \( L \) is equivalent to a language \( L' \) w.r.t the lcp \iff

- \( \bigcap L = \bigcap L' \), and
- for all words \( w \in \Sigma^* \), \( \bigcap(Lw) = \bigcap(L'w) \).

In this case, we write \( L \equiv L' \).

We observe that \( L \) is equivalent to \( L' \) w.r.t. the lcp also after union or concatenation from the left or right with arbitrary other languages. Formally, this amounts to the following properties:

Lemma 6. Assume that \( L, L' \) are non-empty languages and \( L \equiv L' \). Then for every non-empty language \( \hat{L} \), the following holds:

1. \( \bigcap(\hat{L}L) = \bigcap(L'\hat{L}) \);
2. \( \bigcap(\hat{L}L) = \bigcap(\hat{L}L') \);
3. \( \bigcap(L \cup \hat{L}) = \bigcap(L' \cup \hat{L}) \).

Proof. The argument is as follows:

1. \( \bigcap(\hat{L}L) = \bigcap(\bigcap_{w \in \hat{L}}(Lw)) = \bigcap(\bigcap_{w \in \hat{L}}(L'w)) = \bigcap(L'\hat{L}) \);
2. \( \bigcap(\hat{L}L) = \bigcap(\hat{L}\bigcap L) = \bigcap(\hat{L}\bigcap L') = \bigcap(\hat{L}L') \);
3. \( \bigcap(L \cup \hat{L}) = \bigcap L \bigcap \hat{L} = \bigcap L' \bigcap \hat{L} = \bigcap(L' \cup \hat{L}) \).

The next lemma gives us an explicit formula for \( \bigcap(Lw) \) for the special case of the two-element language \( L = \{u, uv\} \).

Lemma 7. Assume that \( u, v \in \Sigma^* \) with \( v \neq \epsilon \). For all words \( w \in \Sigma^* \), \( \bigcap(\{u, uv\}w) = u(w \cap v^\omega) \) holds.

Proof. \( \bigcap(\{u, uv\}w) = uw \cap uvw \). If \( w \) and \( v \) are incomparable or \( w \) is a prefix of \( v \), \( w \cap uvw = w \cap v = w \cap v^\omega \), and the claim follows. Thus, it remains to consider the case that \( v \sqsubseteq w \). Then \( w = v^i w' \) for some \( i \) so that \( v \) is no longer a prefix of \( w' \). Then

\( \bigcap(\{u, uv\}w) = \bigcap(\{u, uv\}v^i w') = uv^i (w' \cap v^\omega) = uv^i (w' \cap v^\omega) = u(w \cap v^\omega) \).
The explicit formula from Lemma 7 can be used to identify small equivalent sublanguages.

**Theorem 3.** For every non-empty language \( L \subseteq \Sigma^* \) there is a language \( L' \subseteq L \) consisting of at most three words such that \( L \equiv L' \).

**Proof.** If \( L \) is a singleton language, we choose \( L' = L \).

Now assume that \( L \) contains at least two words with \( \text{lcp} \ u \). If the \( \text{lcp} \ u \) of \( L \) is not contained in \( L \) then we choose \( L' \) as consisting of the two minimal words \( w_1, w_2 \) so that \( u = w_1 \cap w_2 \). It remains to consider the case where the \( \text{lcp} \ u \) of \( L \) is contained in \( L \). Then we have for each word \( w \in \Sigma^* \),

\[
\bigcap \{Lw\} = \bigcap \{\{uv \mid uw \in L\}\} \\
= \bigcap \{\bigcap \{\{u, uv\}w \mid uw \in L, v \neq \epsilon\}\} \\
= \bigcap \{\{u(w \cap v^\omega) \mid v \in L, v \neq \epsilon\}\} (\text{Lemma 7}) \\
= u(w \cap \bigcap \{v^\omega \mid v \in L, v \neq \epsilon\})
\]

If \( L \) is quasi-periodic, then all words in \( L \) are of the form \( uv_i^\omega \) for some \( v_0 \in \Sigma^+ \) and \( i \geq 0 \) where \( (v_0^\omega)^\omega = v_0^\omega \). Thus, \( \bigcap \{Lw\} = u(w \cap v^\omega) \) for any \( uv \in L \) with \( v \neq \epsilon \). Hence, \( L \equiv L' = \{u, uv\} \) for any such \( v \).

If \( L \) is not quasi-periodic, then we choose words \( w_1, w_2 \in L \) so that the \( \text{lcp} \) of \( v_1^\omega \) and \( v_2^\omega \) has minimal length. Then

\[
\bigcap \{u, uv_1, uv_2\}w = u(w \cap v_1^\omega \cap v_2^\omega) \\
= u(w \cap \bigcap \{v^\omega \mid v \in L, v \neq \epsilon\})
\]

by the minimality of \( v_1 \cap v_2 \). Therefore, \( L \equiv L' = \{u, uv_1, uv_2\} \).

Since for any non-empty words \( w_1, w_2 \) given by SLPs, an SLP for \( w_1^\omega \cap w_2^\omega \) can be computed in polynomial time, we have:

**Corollary 2.** For every non-empty finite \( L \subseteq \Sigma^* \) consisting of words each of which is represented by an SLP, a subset \( L' \subseteq L \) consisting of at most three words can be calculated in polynomial time such that \( L \equiv L' \).

**Proof.** The proof distinguishes the same cases as in the proof of Theorem 3. If \( L \) is a singleton or contains at most three words we are done. Since the words in \( L \) are given as SLPs, we can calculate the \( \text{lcp} \ u \) of the words in \( L \). Next, we determine whether \( u \) is a prefix of every word in \( L \). This can again be checked in polynomial time. If this is not the case, then we can select two words \( w_1, w_2 \in L \) so that \( u = w_1 \cap w_2 \) giving us \( L' = \{w_1, w_2\} \) in polynomial time. So, now assume that \( u \) is a prefix of all words in \( L \). Next, we check whether or not \( L \) is quasi-periodic, i.e., whether for any non-empty words \( v_1, v_2 \) with \( w_1, w_2 \in L \), \( v_1^\omega = v_2^\omega \). By the periodicity lemma of Fine and Wilf (see also Corollary 1), this is the case if \( v_1v_2 = v_2v_1 \). The latter can be checked in polynomial time. If this is the case, then we obtain \( L' = \{u, uv\} \) for some \( uv \in L \) with \( v \neq \epsilon \) in polynomial time.
It remains to consider the case where the lcp $u$ is contained in $L$ and $L$ is not quasi-periodic. Then we need to determine words $w_1$ and $w_2$ in $L$ with $v_1 \neq \epsilon \neq v_2$ such that $v_1^\dagger \sqcap v_2^\dagger$ has minimal length. Since (again by the periodicity lemma of Fine and Wilf) $v_1^\dagger \sqcap v_2^\dagger = v_1 v_2 \sqcap v_2 v_1$, such a pair can be computed in polynomial time as well. Therefore, $L' = \{u, uv_1, uv_2\}$ can be computed in polynomial time.

The following lemma explains that equivalence of two non-empty languages of cardinalities at most 3 can be decided in polynomial time.

**Lemma 8.** Let $L_1, L_2 \subseteq \Sigma^*$ denote non-empty languages consisting of at most three words each, which are all given by SLPs. Then it can be decided in polynomial time whether or not $L_1 \equiv L_2$.

**Proof.** If one of the two languages, contains just a single word, then $L_1 \equiv L_2$ iff $L_1 = L_2$ — which can be decided in polynomial time. Otherwise, we first compute the lcp of $L_1$ and $L_2$, respectively. If these differ, then $L_1$ cannot be equivalent to $L_2$. Therefore assume now that $u$ is both the lcp of $L_1$ and $L_2$, respectively.

Now assume that for all words $v_1, v_2 \in L_1, w_1 \in L_2$ with $v_1 \neq \epsilon \neq v_2$, $v_1^\dagger = v_2^\dagger$. In this case, both $L_1$ and $L_2$ are quasi-periodic with the same period and thus equivalent. Again according to the periodicity lemma of Fine and Wilf, this can be checked in polynomial time.

Next assume that neither $L_1$ nor $L_2$ are quasi-periodic, i.e., there are $v_1, v_2' \in L_1$, $v_1 \neq \epsilon \neq v_2'$, such that $w_i = v_i^\dagger \sqcap v_i'^\dagger$ is minimal for $L_i$ $(i = 1, 2)$. Then $L_1 \equiv L_2$ iff $w_1 = w_2$. Since $w_1, w_2$ can be computed in polynomial time, the result follows in this case as well.

Finally, when none of the listed cases applies, $L_1$ is necessarily inequivalent to $L_2$. Thus, we ultimately arrive at a polynomial time decision procedure for equivalence.

**Remark 1.** Note that in light of the equivalence test, we can choose distinct letters $a, b \in \Sigma$, and equivalently replace the language $L_1 = \{v_1, v_2\}$ with $L_1' = \{u a, u b\}$ whenever $v_1 \neq \epsilon \neq v_2$ and $v_1 \sqcap v_2 = \epsilon$, and the language $L_2 = \{u, uv_1, uv_2\}$ by the language $L_2' = \{u, u a, u b\}$ whenever $w = v_1 v_2 \sqcap v_2 v_1$ does not hold. This kind of reduced representation may allow to use shorter words.

Now we have all pre-requisites to prove the main theorem of our paper.

**Theorem 4.** Assume that $G$ is a context-free grammar with $L = L(G)$ non-empty. Then the longest common prefix of $L$ can be calculated in polynomial time.

**Proof.** Assume w.l.o.g. that $G$ is a CFG in Chomsky normal form. Then we calculate $\sqcap L(G)$ as follows.

We build (implicitly, see the following remark) an acyclic CFG $\hat{G}$ in polynomial time such that $L(\hat{G})$ consists of all words of $L(G)$ for which there is a derivation tree of height at most $4N$ where $N$ is the number of nonterminals in $G$. To this end, for every rewriting rule $A \rightarrow BC$ of $G$ and every $i \in \{1, \ldots, 4N\}$ we add to $\hat{G}$ the rule $A^{(i)} \rightarrow B^{(i-1)}C^{(i-1)}$, and for every rule $A \rightarrow a$ of $G$ we add the rule $A^{(0)} \rightarrow a$ to $\hat{G}$. A straight-forward
induction on \(i\) shows that the derivation trees rooted at \(A^{(i)}\) are isomorphic to the derivation trees rooted at \(A\) of height at most \(i\). For more details, see e.g. [4].

By Theorem 2 we know that \(\bigcap L(G) = \bigcap L(\hat{G})\). By construction, \(\hat{G}\) is also in Chomsky normal form. For \(i\) from 0 to (at most) \(4N\), we then compute in every iteration for every nonterminal \(A^{(i)}\) first the language

\[
[A^{(i)}]' := \{a \in \Sigma^* \mid A^{(0)} \to a \in P\} \cup \bigcup_{A \to BC \in G} [B^{(i-1)}] \cdot [C^{(i-1)}]
\]

By induction on \(i\), we may assume that the languages \([B^{(i-1)}], [C^{(i-1)}]\) (a) have already been computed, (b) consist of at most three words, and (c) every word is given as an SLP. Note that we can drop the assumption that the grammars \(G\) are in Chomsky normal form if the right-hand sides of all rules have bounded lengths.

By virtue of Corollary 2 we therefore can reduce \([A^{(i)}]'\) in polynomial time to a language \([A^{(i)}] \subseteq [A^{(i)}]'\) with \([A^{(i)}] \equiv [A^{(i)}]'\) and \(|[A^{(i)}]| \leq 3\). By construction, we then have

\[
[A^{(i)}] := \{w \in \Sigma^* \mid A^{(i)} \Rightarrow^* w\}
\]

Since \(\hat{G}\) has polynomially many nonterminals only, the overall algorithm runs in polynomial time.

\[\square\]

**Remark 2.** Note that we can drop the assumption that the grammars \(G\) and likewise \(\hat{G}\) are in Chomsky normal form if the right-hand sides of all rules have bounded lengths. Then the cardinality of the languages \([A^{(i)}]'\) are still polynomial. Further, instead of spelling out the grammar \(\hat{G}\) explicitly, we may perform a round robin fixpoint iteration where in every round we first compute

\[
[A]' := \bigcup_{A \to w_1B_1w_2B_2\ldots w_kB_kw_{k+1}} \{w_1\} \cdot [B_1] \cdot \{w_2\} \cdot [B_2] \cdots \{w_k\} \cdot [B_k] \cdot \{w_{k+1}\}
\]

with initially \([A] := \{w \in \Sigma^* \mid A \to w \in G\}\), then updating \([A]\) so that \([A] \subseteq [A]'\) with \([A] \equiv [A]'\) and \(|[A]| \leq 3\). Theorem 2 guarantees that the lcp is attained after at most \(4N\) iterations. Using standard approaches like work lists, we only need to recompute \([A]\) if there is some rule \(A \to \gamma B\delta\) in \(G\) and \([B]\) has changed since the last recomputation of \([A]\). As shown in Lemma 3 we can easily check if \([B] \neq [B]'\) in every round and accordingly insert \(A\) into the work list.

We demonstrate this simplified version of the algorithm described in Theorem 3 by an example.

**Example 3.** Consider the following grammar \(G\) with the following rules:

\[
\begin{align*}
S &\to \text{Aababaac} \\
A &\to \text{ab Aabaab} \mid \text{ab Aabaac} \mid \epsilon
\end{align*}
\]

The round robin fixpoint iteration would proceed by iteratively evaluating the equations

\[
[A]' := \{abwabaab, abwabaac, \varepsilon \mid w \in [A]\}
\]

\[
[S]' := \{wababaac \mid w \in [A]\}
\]
and recomputing the languages $[A]$ and $[S]$ so that $[A] \equiv [A]'$ and $[S] \equiv [S]'$ and both $[A]$ and $[S]$ consist of at most three words where we further reduce the words of $[A]$ and $[S]$ as described in the remark following Lemma 8. As $[A]$ does not depend on $[S]$, we can postpone the computation of $[S]$ after $[A]$ has converged. In the first round, we have:

$$[A] = [A]' = \{\epsilon\}$$

For the second round, we first calculate:

$$[A]' = ab\{\epsilon\}abaab \cup ab\{\epsilon\}abaac \cup \{\epsilon\} = \{ababaab, ababaac, \epsilon\}$$

and thus update $[A]$ to $[A] := \{(ab)^2aab, (ab)^2aac, \epsilon\}$. For the third round, we obtain

$$[A]' = ab\{(ab)^2aab, (ab)^2aac, \epsilon\}abaab \cup ab\{(ab)^2aab, (ab)^2aac, \epsilon\}abaac \cup \{\epsilon\}$$

$$= \{(ab)^3a(ab)^3aab, (ab)^3aacabab, (ab)^3aab\} \cup \{(ab)^3a(ab)^3aac, (ab)^3aadabaac, (ab)^2aac\} \cup \{\epsilon\}$$

$$\equiv \{(ab)^3aababaab, (ab)^3aab, \epsilon\}$$

$$\equiv \{(ab)^3, (ab)^2aa, \epsilon\}$$

$$=: [A]$$

which is already the fixpoint. Therefore we obtain

$$[S]' = \{(ab)^3, (ab)^2aa, \epsilon\}ababaac$$

$$= \{(ab)^3(ab)^2aac, (ab)^2aa(ab)^2aac, (ab)^2aac\}$$

$$\equiv \{(ab)^3(ab)^2aac, (ab)^2aac\}$$

$$\equiv \{(ab)^3, (ab)^2aa\}$$

$$=: [S]$$

So $L = \{(ab)^3 \cap (ab)^2aa = (ab)^2a$.}

5 Conclusion

We have shown that the longest common prefix of a non-empty context-free language can be computed in polynomial time. This result was based on two structural results, namely, that it suffices to consider words with derivation trees of bounded height, and second that each non-empty language is equivalent to a sublanguage consisting of at most three elements. For the actual algorithm, we relied on succinct representations of long words by means of SLPs. It remains as an intriguing open question whether the presented method can be generalized to more expressive grammar formalisms.

References

[1] 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.
[2] Adrien Boiret and Raphaela Palenta. Deciding equivalence of linear tree-to-word transducers in polynomial time. In Sreˇ cko Brlek and Christophe Reutenauer, editors, Developments in Language Theory: 20th International Conference, DLT 2016, pages 355–367. LNCS 9840, Springer, 2016.

[3] Christian Choffrut and Juhani Karhumäki. Combinatorics of Words, pages 329–438. Springer Berlin Heidelberg, 1997.

[4] Javier Esparza, Michael Luttenberger, and Maximilian Schlund. A brief history of strahler numbers. In Language and Automata Theory and Applications - 8th International Conference, LATA 2014, pages 1–13. LNCS 8370, Springer, 2014.

[5] N. J. Fine and H. S. Wilf. Uniqueness theorems for periodic functions. Proceedings of the American Mathematical Society, 16(1):109–114, 1965.

[6] Grégoire Laurence, Aurélien Lemay, Joachim Niehren, Sławek Staworko, and Marc Tommasi. Normalization of sequential top-down tree-to-word transducers. In Adrian-Horia Dediu, Shunsuke Inenaga, and Carlos Martín-Vide, editors, Language and Automata Theory and Applications: 5th International Conference, LATA 2011, pages 354–365. LNCS 6638, Springer, 2011.

[7] Grégoire Laurence, Aurélien Lemay, Joachim Niehren, Sławek Staworko, and Marc Tommasi. Learning sequential tree-to-word transducers. In Adrian-Horia Dediu, Carlos Martín-Vide, José-Luis Sierra-Rodríguez, and Bianca Truthe, editors, Language and Automata Theory and Applications: 8th International Conference, LATA 2014, pages 490–502. LNCS 8370, Springer, 2014.

[8] Markus Lohrey. The Compressed Word Problem for Groups. Springer Briefs in Mathematics. Springer, 2014.

[9] Wojciech Plandowski. Testing equivalence of morphisms on context-free languages. In Algorithms - ESA ’94, Second Annual European Symposium, pages 460–470. LNCS 855, Springer, 1994.

6 Appendix

6.1 Proof of Theorem [1]

We split the proof of the theorem into several lemmata covering the cases

1. $L = (x, \varepsilon)(y, \varepsilon) + (z, \varepsilon)^* w = x(y + z)^* w$ (cf. Lemma [9])

2. $L = (x, \bar{x})(y, \bar{y})^* w$ (cf. Lemma [5])

3. $L = (x, \bar{x})(y, \bar{y}) + (z, \bar{z})^* w$ (cf. Lemma [10]), and

4. $L = (x, \bar{x})(y_1, \bar{y}_1) + \ldots + (y_l, \bar{y}_l)^* w$ for arbitrary $l \in \mathbb{N}$. (cf. Lemma [11]).
Lemma 9. $L = x(y + z)^*w \Rightarrow \bigcap L = x \bigcap (y + z)^{\leq 1}w$

Proof. As $x$ does not matter, simply assume $x = \varepsilon$. We show by induction on $m$ that for any $\alpha \in (y + z)^{m}w$

$$w \cap yw \cap zw = w \cap yw \cap zw \cap \alpha$$

The case $m \leq 1$ is obviously true. Fix any $m > 1$ and any $\alpha \in (y + z)^{m+1}w$; wlog. $\alpha = \alpha'yw$. Set $w' = w \cap yw$. Then:

$$w \cap yw \cap zw \overset{\text{Induction}}{=} w \cap yw \cap zw \cap \alpha'w$$

$$= w' \cap zw \cap \alpha'w'$$

$$= w' \cap zw \cap \alpha'(w \cap yw)$$

$$= w \cap yw \cap zw \cap \alpha'w \cap \alpha'yw$$

\hfill □

Lemma 10. Let $L = (x, \bar{x}][(y, \bar{y}) + (z, \bar{z})]^*w$. Then $\bigcap L = \bigcap ((x, \bar{x})][(y, \bar{y})^{\leq 2} + (z, \bar{z})^{\leq 2}]w$.

Proof. The case $y = \varepsilon = z$, i.e. $L = xw(\bar{y} + \bar{z})^*\bar{x}$ is already proven in Lemma 9.

Consider the case $yz \neq zy$. Let $\alpha$ be a witness (w.r.t. $xw\bar{x}$). Assume $\alpha$ is of the following form for some suitable $j \geq 0$

$$\alpha = xy^jyz\alpha'\bar{z}y\bar{y}^j\bar{x}$$

The case $\alpha = xz^jyz\alpha'\bar{z}y\bar{y}^j\bar{x}$ is symmetrical.) Then (swapping the inner most $y$ and $z$ still yields a word of $L$):

$$\bigcap L = xw\bar{x} \cap \alpha = xw\bar{x} \cap xy^jyz\alpha'\bar{z}y\bar{y}^j\bar{x} \cap xy^jz^jy\alpha'\bar{z}y\bar{y}^j\bar{x} = xw\bar{x} \cap xy^j(yz \cap zy)$$

Using Corollary 1

$$xw\bar{x} \cap xy^j(yz \cap zy) = xw\bar{x} \cap xy^j(y^\omega \cap z^\omega)$$

But obviously

$$xyzw\bar{y}x \cap xyzw\bar{y}x = x(yz \cap zy) \subseteq xy^j(yz \cap zy)$$

So either $xyzw\bar{y}x$ or $xyzw\bar{y}x$ has to be a witness, too.

But again by virtue of Corollary 1 and for sufficiently large $j$

$$xyzw\bar{y}x \cap xyzw\bar{y}x = xy^\omega \cap xz^\omega = (x, \bar{x})(y, \bar{y})^jw \cap (x, \bar{x})(z, \bar{z})^jw$$

Hence, we already find a witness within $(x, \bar{x})][(y, \bar{y})^* + (z, \bar{z})^*]w$ and, thus, within $(x, \bar{x})][(y, \bar{y})^{\leq 2} + (z, \bar{z})^{\leq 2}]w$

So assume for the following that $yz = zy$ with $y = p^k \land z = p^l$ and $p$ primitive (wlog. $k \geq l$). Wlog. $\max\{k, l\} > 0$. (If $k = 0$, then $y = \varepsilon = z$ which we have already discussed.)

Let $w = p^mp'w'$ with $p = p^mp''$ and $w \cap p'w = w \cap p^{m+1} = p^mp' \subset p^{m+1}$. Set $q = p''p'$ s.t. $pp' = p'q$ and $q \cap w' = \varepsilon$ and $p'' \neq \varepsilon$. As $p$ is primitive, so is $q$.

If $w \not\sqsubset p''$, then $w' \neq \varepsilon$. If $y \neq \varepsilon$, then $xyw\bar{y}x$ is a witness; if $z \neq \varepsilon$, then $zxyw\bar{y}x$ is a witness, too.

Hence, $w \subset p''$ in the following. Then $w' = \varepsilon$ and $pw = wq$. We factorize $\bar{x}, \bar{y}, \bar{z}$ w.r.t. $q$. 

16
• Let \( x \cap q^w = q^n q' \sqsubset q^{n+1} \) with \( x = q^n q' \sqsetminus x' \) and \( q = q' q'' \) and \( q'' \neq \varepsilon \).

• Let \( y \cap q^w = q^l q' \sqsubset q^{l+1} \) with \( y = q^l q y' \) and \( q = q y' q'' \) and \( \hat{q} \neq \varepsilon \).

• Let \( z \cap q^w = q^l q' \sqsubset q^{l+1} \) with \( z = q^l q \hat{z}' \) and \( q = \hat{q} q \) and \( \hat{q} \neq \varepsilon \).

Thus:

\[
L = (xw, q^n q' x')[(q^k, q^k \hat{y} y') + (q^l, q^l q^\varepsilon)]^* \varepsilon
\]

Wlog. \( y = p^k \neq \varepsilon \), i.e. \( k > 0 \), and further \( k \geq l \).

Let \( \alpha \) be a witness w.r.t. \( x w x \). We may distinguish the following cases for a witness \( \alpha \) (with \( \Gamma = (y, y) + (z, z) \)):

1. \( \alpha \in (x, x)(y, y) w \)
2. \( \alpha \in (x, x)(z, z) w \)
3. \( \alpha \in (x, x) \Gamma^*(z, z)(y, y) w \)
4. \( \alpha \in (x, x) \Gamma^*(y, y) (z, z) w \)

(The case \( \alpha = x w x \) is covered by both (1) and (2).)

Cases (1) and (2) are both covered by Lemma \[13\]. Hence, we may assume in the following that any witness is of the form (3) or (4) – otherwise we are done.

As \( k > 0 \), we have

\[
\bigcap L \subseteq x w x \sqcap (x, x)(y, y) w = x w (x \sqcap q^{k(n+1)}) = x w (x \sqcap q^{n+1}) = x w q^n q'
\]

Hence \( \bigcap L = x w \phi \) for some suitable \( \phi \sqsubset q^n q' \); if \( \phi = q^n q' \), then \( (x, x)(y, y) w \) would be a witness, contradicting our assumption that any witness of the form (3) or (4).

Hence, \( \phi \sqsubset q^n q' \), i.e. \( \bigcap L = x w \phi \sqsubset x w q^n q' = x w (x \sqcap q^n) \); so, any witness \( \alpha \) has to satisfy

\[
x w \phi = x w x \sqcap \alpha = x w q^n q' \sqcap \alpha = x w q^n \sqcap \alpha
\]

as \( \alpha \) has to differ from \( x w x \) within the suffix \( q^n q' \).

If \( n < k \), then

\[
x w x \sqcap x \ldots y \ldots w \ldots z \ldots \hat{z} = x w (x \sqcap q^k) = x w (x \sqcap q^{n+1}) = x w q^n q'
\]

So, either \( x y w y \hat{x} \) is a witness, or \( y \) cannot occur in any witness, implying that either \( x z w x \hat{x} \) or \( x z z w x \hat{x} \) is a witness.

Hence, \( n \geq k \geq l \) with \( n > 0 \) as \( k > 0 \).

We need to take a closer look at the structure of a respective \( \alpha \). As case (4) is a special case of (3), we discuss (3) in detail and only remark where the proof differs from case (4).

So assume \( \alpha \in (x, x) \Gamma^*(z, z)(y, y) w \). Then

\[
\alpha = x w q^{k+1} q^k q' q'' y q y' q'' q' y' q'' \beta q^n q' x'
\]

\[17\]
where \( \lambda \) (\( \mu \)) is the number of \( z \) (\( y \)) right of \( x \) and left of the inner most \( z \); and \( \beta \) is the corresponding string of \( \hat{y}, \hat{z} \), e.g. if \( \lambda + \mu = 0 \), then \( \beta = \varepsilon \).

As \( \alpha \cap xw\bar{x} = xw\phi \sqsubset xwq^nq' \), we have
\[
q^kq'^k \hat{q} \sqsubseteq q^lq'^k\hat{q} \sqsubseteq q^{\lambda + \mu k}q'^k \hat{q} \sqsubseteq \phi \sqsubset q^nq'
\]

If \( \hat{y}' \neq \varepsilon \), then
\[
xw\phi = \alpha \cap xwq^n = xwq^\lambda q^k q'^k \hat{q} \sqsubseteq xw\bar{x} \cap xyw\bar{y}x = xw(q^n \cap q^k \hat{q}) = xwq^n \hat{q}
\]

So \( \hat{y}' = \varepsilon \) in the following.

We first do away with the case \( \hat{q} = \varepsilon \):

If \( \hat{q} = \varepsilon \) (i.e. \( \hat{y} = q^k \)), then
\[
\alpha = xwq^{\lambda + \mu k}q'^k \hat{q} \sqsubseteq xwq^n \hat{q}
\]

Hence:
\[
q^lq'^k \hat{q} \sqsubseteq q^lq'^k \hat{q} \sqsubseteq q^{\lambda + \mu k}q'^k \hat{q} \sqsubseteq q^nq'
\]

**Note:** As \( k > 0 \), we trivially have \( qq^n \hat{q} \sqsubseteq q^{\lambda + \mu k}q'^k \hat{q} \) in contrast to case (4); but we always may assume in the case \( q \neq q^n \) as otherwise \( \hat{y}' = \varepsilon \) which also holds analogously in case (4).

Obviously \( \hat{z}' \neq \varepsilon \), as otherwise \( xw\bar{x} \cap \alpha = xwq^nq' \).

If \( \hat{z}' \neq \varepsilon \), we obtain the contradiction \( xw\bar{x} \cap \alpha \sqsubseteq xw\bar{x} \cap xzw\bar{z}x = xwq^l \hat{q} \) as \( q^l \hat{q} \sqsubseteq q^nq' \); so \( \hat{q} \neq \varepsilon = \hat{z}' \) (i.e. \( \hat{z} = q^l \hat{q} \)), and \( \beta \in (q^nq')^* \).

As \( q \) primitive and \( \varepsilon \neq q \sqsubset q \), we have \( \hat{q} \neq q \hat{q} \) and thus \( \hat{z}' \cap q^n = \hat{q} \cap q \hat{q} \sqsubset q \).

Hence (using \( n > 0 \) and \( qq^n \hat{q} \sqsubseteq q^n \hat{q} \))
\[
xw\phi \sqsubset xw\bar{x} \cap xzw\bar{z}x = xw(q^n \hat{q} \sqcup q^{l+q} \hat{q}) = xwq^{l+q}(q \cap \hat{q}) \sqcup xwq^{l+q} \hat{q} \sqsubseteq xw\phi
\]

Thus also \( \hat{q} \neq \varepsilon \) from here on.

Again, as \( q \) primitive and \( \varepsilon \neq q \sqsubset q \), we have \( \hat{q} \neq q \hat{q} \) and thus \( \hat{z}' \cap q^n = \hat{q} \cap q \hat{q} \sqsubset q \).

Hence (using \( n > 0 \))
\[
xw\phi \sqsubset xw\bar{x} \cap xyw\bar{y}x = xw(q^n \hat{q} \sqcup q^{l+q} \hat{q}) \sqsubset xwq^{k+q} \hat{q}
\]

If \( \lambda + \mu k + l + jk > 0 \), we obtain the contradiction
\[
qq^{k+q} \hat{q} \sqsubseteq \phi \sqsubset qq^{k+q} \hat{q}
\]

analogously to the case \( \hat{y}' = \varepsilon \).

**Note:** In case (4) we are done at this point, as \( k > 0 \) takes the place of \( l \geq 0 \) in case (4).

So \( \lambda + \mu k + l + jk = 0 \), i.e. \( l = j = 0 \) as \( k > 0 \) for the following allowing us to write
\[
\alpha = xwq^{k+q} \hat{q} \sqsubseteq \phi \sqsubset qq^{k+q} \hat{q}
\]
If \( l' > 0 \), then (using \( n > 0 \) and \( \hat{q} \sqsubseteq q \))

\[
xw\phi = \alpha \sqcap xwq^\omega \sqcap xyw\bar{y}x \sqcap xwq^\omega = xw(q^{k+k'}\hat{q}q \cap q^\omega) = \alpha \sqcap xwq^\omega
\]

So we have to have \( l' = 0 \) which allows us to further simplify \( \alpha \):

\[
\alpha = xwq^kq^{k'}\hat{q}qz'(q\bar{z}')^\lambda q^nq'\bar{x}'
\]

If \( \bar{z}' \neq \varepsilon \), then (using \( n > 0 \) and \( \hat{q} \sqsubseteq q \))

\[
xw\phi \sqcap xw\bar{x} \sqcap xzw\bar{z}x = xw(q^nq' \sqcap xwq\bar{z}'q^nq') = xw\bar{q} \sqcap xwq \sqsubseteq xwq^kq^{k'}\hat{q} \sqsubseteq xw\phi
\]

So \( \bar{z}' = \varepsilon \) and thus \( \hat{q} \neq \varepsilon \) (else \( z = \varepsilon \) and \( \bar{z} = \varepsilon \)):

\[
\alpha = xwq^kq^{k'}\hat{q}q(q\bar{q})^\lambda q^nq'\bar{x}'
\]

Again we then have \( \bar{q}q \neq q\bar{q} \), i.e. \( q^\omega \sqcap \bar{q}^\omega = q\bar{q} \sqcap \bar{q}q \sqsubset q\bar{q} \).

Hence (using \( n > 0 \) and \( \hat{q} \sqsubseteq q \)):

\[
xw\bar{x} \sqcap xzw\bar{z}x = xw(q^nq' \sqcap q\bar{q}^nq') \sqcap xw\bar{q}.
\]

We therefore have

\[
xwq^{k+k'}\hat{q} \sqsubseteq \alpha \sqcap xwq^\omega = xw\phi \sqcap xw\bar{q}
\]

i.e. \( k' = 0, k = 1 \), and \( \hat{q} \sqsubseteq \hat{q} \) s.t.:

\[
\alpha = xwq\bar{q}q(q\bar{q})^\lambda q^nq'\bar{x}'
\]

As \( n > 0 \), \( \hat{q} \sqsubseteq \hat{q} \sqsubseteq q \), \( \phi \sqsubset q\bar{q} \), we obtain the final contradiction:

\[
xw\phi = \alpha \sqcap xwq^\omega \mid \phi < [q\bar{q}] \leq [q\bar{q}] \quad xw(q\bar{q}q \sqcap q^nq') = xw(q\bar{q}q \sqcap q^nq') \sqcap xw\bar{q}\bar{x} \sqcap xwq^\omega
\]

\[\square\]

**Lemma 11.** Let \( L = (x, \bar{x})[\sum_{i=1}^n(y_i, \bar{y}_i)]^*w \). Then \( \sqcap L \sqcap L = \sqcap \{(x, \bar{x})[\sum_{i=1}^n(y_i, \bar{y}_i)] \leq 2\}w \).

**Proof.** Let \( \alpha \in L \) be a witness i.e. \( \sqcap L = xw\bar{x} \sqcap \alpha \).

Then

\[
\alpha = (x, \bar{x}) \prod_{j=1}^k (y_{i_j}, \bar{y}_{i_j})w
\]

for suitable \( i_1, \ldots, i_k \in \{1, \ldots, n\} \).

If \( k = 1 \), we are done. Assume \( k \geq 2 \) for any witness (and any such factorization).

Pick a witness \( \alpha \) and a factorization that minimizes \( k \). Set

\[
(y, \bar{y}) = (y_{i_1}, \bar{y}_{i_1}) \quad (z, \bar{z}) = (y_{i_2}, \bar{y}_{i_2}) \quad w' = \prod_{j=3}^k (y_{i_j}, \bar{y}_{i_j})w
\]

Then using Lemma [10]

\[
\sqcap L = xw\bar{x} \sqcap \alpha = xw\bar{x} \sqcap \left[ (x, \bar{x})[(y, \bar{y}) + (z, \bar{z})]^*w' = xw\bar{x} \sqcap \left[ (x, \bar{x})[(y, \bar{y}) \leq 2 + (z, \bar{z}) \leq 2]w' \right] \sqcap \sqcap \right]_{\sqsubseteq L \sqcap \sqcap \alpha}
\]

19
So, one of the words on the right-hand side has to be a witness too. If \( k = 2 \), we have \( w' = w \), and we are also done. Hence assume \( k \geq 3 \) from now on. Because of our assumption that \( \alpha \) is a witness with a minimal factorization, only \((x, \bar{x})(y, \bar{y})^2 w' \) or \((x, \bar{x})(z, \bar{z})^2 w' \) can be a witness. Because of symmetry, it suffices to assume \( \cap L = xw\bar{x} \cap (x, \bar{x})(y, \bar{y})^2 w' \). As \( k \geq 3 \) we have \( w' = (y_{i_3}, \bar{y}_{i_3}) \prod_{j=4}^{k} (y_{i_j}, \bar{y}_{i_j}) w \). Set \((z', \bar{z}') = (y_{i_3}, \bar{y}_{i_3}) \) and \( w'' = \prod_{j=4}^{k} (y_{i_j}, \bar{y}_{i_j}) w \) so that

\[
\cap L = xw\bar{x} \cap (x, \bar{x})((y, \bar{y})^2 + (z', \bar{z}'))^* w'' \subseteq L
\]

Using again Lemma \([10]\) this is equivalent to

\[
\cap L = xw\bar{x} \cap xw'' \bar{x} \cap (x, \bar{x})(z', \bar{z}')^* w'' \cap (x, \bar{x})(y, \bar{y})^4 w'' \cap (x, \bar{x})(z', \bar{z}'')^* w''
\]

As \((x, \bar{x})(y, \bar{y})^* w' \subseteq L\), adding \( \cap (x, \bar{x})(y, \bar{y})^* w'' \) cannot change the lcp

\[
\cap L = xw\bar{x} \cap xw'' \bar{x} \cap (x, \bar{x})(z', \bar{z}')^* w'' \cap (x, \bar{x})(z', \bar{z}'')^* w'' \cap \cap (x, \bar{x})(y, \bar{y})^* w''
\]

Using Lemma \([5]\) we finally obtain

\[
\cap L = xw\bar{x} \cap xw'' \bar{x} \cap (x, \bar{x})(z', \bar{z}')^* w'' \cap (x, \bar{x})(z', \bar{z}'')^* w'' \cap \cap (x, \bar{x})(y, \bar{y})^* w''
\]

Again, we have to find another witness within the words occurring on the right-hand side. But all these words have a factorization using less factors than \( \alpha \) contradicting our choice of \( \alpha \). Hence, there has to be a witness having a factorization with \( k \leq 2 \). Thus:

\[
\cap L = \cap_{i=1}^{n} (x, \bar{x})(y_i, \bar{y}_i)^2 w
\]