Characterizing Non-Regularity

Martin Berglund

Umeå University, Sweden
mbe@cs.umu.se

In collaboration with Henrik Björklund and Frank Drewes.

Abstract. This paper considers a characterization of the context-free non-regular languages, conjecturing that there for all such languages exists a fixed string that can be pumped to exhibit infinitely many equivalence classes. A proof is given only for a special case, but the general statement is conjectured to hold. The conjecture is then shown to imply that the shuffle of two context-free languages is not context-free.

1 Introduction

This paper is concerned with the characterization of context-free languages, a subject with a long and interesting history. The context-free class of languages is after all very important, it is a both large and interesting class for which parsing is reasonably efficient both in theory and practice. See e.g. [HMU03] for the basics on context-free languages. We will assume a passing familiarity with formal language theory and will only recall the most important definitions we need.

Specifically, we will consider a conjecture, considered likely to be true by the author, which characterizes the *non-regular* context-free languages. Consider Figure 1. Where classic characterizations such as Parikh's theorem and the context-free pump-

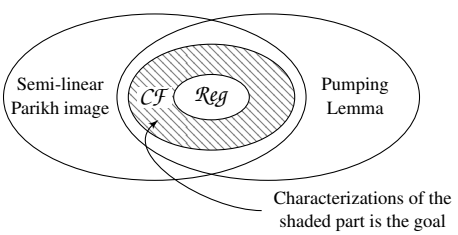

Fig. 1: A Venn diagram illustrating what is being looked at. The set of languages with semi-linear Parikh images are a (strict) superset of the context-free ($\mathcal{CF}$), the languages that fulfill the context-free pumping lemma form a (disjoint and strict) superset of the context-free. The conjecture here studied will restrict itself to just the context-free (by assumption) but will characterize the non-regular (non-$\mathcal{Reg}$) part of the context-free.

ing lemma state requirements that all context-free languages fulfill, this conjecture instead precisely defines which context-free languages are non-regular.

Stating and proving some properties about a conjecture characterizing the inner border of that shaded area, $\mathcal{CF} \setminus \mathcal{Reg}$ in Figure 1, is one part of this paper. The other proves a result that was part of the original motivation for this conjecture, relating to the shuffle. The shuffle of two languages consists of all strings that can be constructed by picking one string from each language and interleaving them. It seems highly probable that the shuffle of any two context-free languages should produce a language which is *not* context-free, *unless* one of those two original context-free languages was actually regular. A more specialized case of this statement will be proven to be true if the conjecture holds.

Overview of the paper. All this will become clearer once some baseline definitions are given, so the intuition about the conjecture is continued in Section 3. In Section 4 the conjecture itself is introduced. In Section 6 the impact it would have on the shuffle of context-free languages if true.

Acknowledgments and citations. The author is indebted to his advisors Henrik Björklund and Frank Drewes for input on the contents of this paper. As most bibliography formats lack a more appropriate field it is suggested that they are listed as author on citations of this paper.

2 Basic Notation

Let $\mathbb{N} = \{0, 1, \ldots\}$. Let ε denote the empty string. Let Σ denote an arbitrary alphabet (finite set of symbols). L^* denotes the Kleene closure (i.e. Σ^* is the language of all strings). We let $w^k = ww^{k-1}$ with $w^0 = \varepsilon$ (i.e. the concatenation of k copies of a string). For a string $w = \alpha_1 \cdots \alpha_n$ (with all $\alpha_i \in \Sigma$) let $w(i) = \alpha_1 \cdots \alpha_i$ and $w(i, j) = \alpha_i \cdots \alpha_j$. As already seen we let $\mathcal{Reg}$ denote the regular languages and $\mathcal{CF}$ denote the context-free languages. We let $\mathcal{L}(A)$ denote the language generated by a grammar/automaton A. $L \propto L'$ denotes the left quotient:

Definition 2 For languages $\mathcal{L}, \mathcal{L}' \subseteq \Sigma^*$ the left quotient $\mathcal{L} \propto \mathcal{L}'$ is defined by letting $w \in \mathcal{L} \propto \mathcal{L}'$ if and only if $vw \in \mathcal{L}'$ for some $v \in \mathcal{L}$. $\diamond$

To simplify the notation in some common cases we, for a string $u \in \Sigma^*$, write just u to mean the language $\{u\}$ in this context. For example $u \propto \mathcal{L}$ and $\mathcal{L} \propto u$ are equivalent to $\{u\} \propto L$ and $\mathcal{L} \propto \{u\}$ respectively. Let $w \equiv_{\mathcal{L}} v$ if and only if $w \propto \mathcal{L} = v \propto \mathcal{L}$, the subclasses of strings induced by $\equiv_{\mathcal{L}}$ are called the *equivalence classes* of $\mathcal{L}$. The relation $\equiv_L$ is also known as the *right congruence* induced by L.

3 Further Introductory Discussion

Among the characterizing results about context-free languages there are few better known than the context-free pumping lemma [BHPS61]. Let us briefly recall it.

Lemma 3 For every context-free language $L \subseteq \Sigma^*$ there exists some constant $k \in \mathbb{N}$ such that every string $w \in L$ with $|w| > k$ can be split into strings $x, v, y, w, z \in \Sigma^*$ (i.e. $w = xvywz$) such that

1. the middle piece is not too long, $|vyw| \leq k$,
2. we can repeat v and w arbitrarily many times, $xv^i yw^i z \in L$ for all $i \in \mathbb{N}$,
3. the repetition part is non-empty, $|vw| > 0$. ◇

As illustrated in Figure 1 this lemma *encapsulates* context-free languages neatly, proving for example that $\{a^n b^n c^n \mid n \in \mathbb{N}\}$ is not context-free is straightforward as requirements 2 and 3 mean at least one a, b or c must be repeatable, but requirement 1 ensures that we cannot fit at least one each of all three symbols into vw without making the middle too long.

However, what this lemma does *not* tell us is (at least) two-fold. Firstly, for every language L the language Laa^* fulfills the lemma. That is, arbitrarily complex languages can fulfill the lemma if you "attach" them to a language that fulfills it. Secondly, and more importantly for our concerns here, the lemma does not tell us if the language is "just" regular. That is, if the lemma is fulfilled using e.g. $w = \varepsilon$ then it is equivalent to the pumping lemma for regular languages. As shown in Figure 1 we are aiming to instead make a statement about the *non-regular* context-free languages.

The characterization through the semi-linear Parikh image [Par66] is of course similar in the first respect, $L\Sigma^*$ has a semi-linear Parikh image for all L, and the statement of the theorem is precisely that context-free and regular languages have the same class of Parikh images.

It is important to note that there of course *are* strict characterizations of the $\mathcal{CF} \setminus \mathcal{Reg}$ class, simply stating "a language L such that a context-free grammar G exists with $\mathcal{L}(G) = L$ but no finite automaton A exists with $\mathcal{L}(A) = L$" is sufficient. The perhaps most used characterization for $\mathcal{CF} \setminus \mathcal{Reg}$ is to make use of the fact that infinitely many equivalence classes exist, in effect using the converse of the statement of the Myhill-Nerode theorem (see e.g. [HMU03]) to characterize non-regularity.

This is however a matter of how *helpful* the statement is. In short, what we are aiming for is the intuition that every non-regular context-free language has a choice of x and v in the pumping lemma (as sketched in Definition 3) for which the set of choices for y, w, z that fulfill the lemma is *non-empty*, but all have $w \neq \varepsilon$. That is, if the language is non-regular then there exists a choice of string to repeat which *must* be matched by a change in the suffix of the language.

Another way of viewing matters is through the Chomsky-Schützenberger theorem [CS63], which *precisely* defines the context-free languages, basically defining an alternative representation alongside the context-free grammars. To avoid recalling this theorem in full we just note that that the intuition corresponding to the conjecture which follows here would, in terms of the Chomsky-Schützenberger theorem, state that the homomorphism may *not* project away the difference between opening and closing parenthesis in the underlying Dyck language.

4 The Characterizing Conjecture

We now arrive at the main topic of this paper, a conjecture which, if true, characterizes the non-regular context-free languages.

Conjecture 4 For $\mathcal{L} \in \mathcal{CF}$ (over Σ) it holds that $\mathcal{L} \in \mathcal{CF} \setminus \mathcal{Reg}$ *if and only if* there exists some $x, v \in \Sigma^*$ such that for all $n, m \in \mathbb{N}$ with $n \neq m$ it holds that $xv^n \not\equiv_{\mathcal{L}} xv^m$ (i.e., that $(xv^n \propto \mathcal{L}) \neq (xv^m \propto \mathcal{L})$). $\diamond$

Hopefully it is already clear that this statement is of a rather fundamental nature. It characterizes the context-free languages without including the regular languages, in a way that the author believes has not otherwise been done. Unsurprisingly the author is also intuitively convinced that the conjecture holds, as it appears to strike at the core of what context-free languages *do*. It is important to note that this statement is not a trivial consequence of the pumping lemma for context-free languages (or Ogden's lemma or other variations), as those characterize all context-free languages, including the regular. In addition it may be instructive to keep in mind that for $L \in \mathcal{CF}$ it is, in general, undecidable whether $L \in \mathcal{Reg}$, which implies that any proof of Conjecture 4 must be non-constructive.

5 Proving a Fragment of Conjecture 4

There are, luckily, some subclasses of the context-free languages for which it is easier to see that Conjecture 4 holds than it is for the full class. It is, of course, very easy to see that it holds for the language $\{a^n b^n \mid n \in \mathbb{N}\}$, by choosing $x = \varepsilon$ and $v = a$. We can, however, broaden this to a much larger (though somewhat related) subclass of the context-free languages still.

First we recall the definition of push-down automata, mainly in order to name to name some restrictions on the language class precisely.

Definition 5 A *push-down automaton* (PDA) is a tuple $A = (Q, \Sigma, \Gamma, \delta, q_0, \bot, F)$, where

- Q is the finite set of states,
- Σ is the input alphabet,
- Γ is the stack alphabet,
- $q_0 \in Q$ is the initial state,
- $F \subseteq Q$ is the set of final states,
- $\bot \in \Gamma$ is the bottom stack symbol, and, finally,
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$ is the finite set of rules. For all rules $(q, \alpha, \gamma, q', s) \in \delta$ where $\gamma = \bot$ it must hold that $s = s'\bot$ for some $s' \in \Gamma^*$.

The set of configurations of A is $C_A = Q \times (\Gamma^* \cdot \bot)$. A can go from the configuration $(q, \gamma s)$ to $(q', s's)$, for $\gamma \in \Gamma$ and $s, s' \in \Gamma^*$, by reading the symbol α (may be ε in which case nothing is read) if and only if there is a rule $(q, \alpha, \gamma, q', s') \in \delta$. The initial configuration is $(q_0, \bot)$. A *accepts* a string w, i.e. $w \in \mathcal{L}(A)$, if and only if the string

can be read going from the configuration $(q_0, \bot)$ to an accepting configuration. For general push-down automata the accepting configurations are all (q,s) such that $q \in F$ and $s \in \Gamma^*$.

If A is a *stack-emptying PDA* the accepting configurations are the set $\{(q,\bot) \mid q \in F\}$. A is *fully deterministic* if δ is a function $\delta : (Q \times \Sigma \times \Gamma) \to (Q \times \Gamma^*)$. $\diamond$

Next let us note a small, and fairly obvious detail, before we use it in the broader proof. Most readers will probably accept it as true without further argument, but for completeness sake the proof is sketched.

Lemma 6 For every language $L \in \mathcal{CF} \setminus \mathcal{Reg}$ there exists an infinite string ω such that $\omega(i) \not\equiv_L \omega(j)$ for all $i \neq j$. $\diamond$

PROOF: This is necessarily the case, since the Myhill-Nerode theorem dictates that $L \notin \mathcal{Reg}$ if and only if L has infinitely many equivalence classes. Simply construct the tree where the root is marked ε, and for each node v give it a child for each $\alpha \in \Sigma$ such that $v\alpha \not\equiv_L v(i)$ for every i. Each node has a finite number of children, but the infinitely many equivalence classes will necessarily be reached in this fashion, so by Königs lemma some path is infinite, which we take as ω. ∎

Theorem 7 If $\mathcal{L}$ is a language accepted by a stack-emptying fully deterministic PDA A then Conjecture 4 holds for $\mathcal{L}$. $\diamond$

PROOF: Clearly, if $\mathcal{L} \in \mathcal{Reg}$ Conjecture 4 holds, since $\mathcal{L}$ has a finite set of equivalence classes according to the Myhill-Nerode theorem, and so for all $x, v \in \Sigma^*$ there must exist $i \neq j$ with $xv^i \equiv_{\mathcal{L}} xv^j$.

The case where $\mathcal{L} \in \mathcal{CF} \setminus \mathcal{Reg}$ remains. Let ω be an infinite word such that $\omega(i) \not\equiv_{\mathcal{L}} \omega(j)$ for all $i, j \in \mathbb{N}$. Let (q_i, s_i) be the configuration A reaches on $\omega(i)$. There will be precisely one since A is fully deterministic. Then let $S : \mathbb{N} \to Q \times \Gamma$ be such that $S(i) = (q_i, \gamma_i)$ where $s_i = \gamma_i s'_i$ for some $\gamma \in \Gamma$. Let $H(i) = |s_i|$ (i.e. the height of the stack of the configuration reached on $\omega(i)$).

All (q_i, s_i) must be different, by the construction of ω and the fact that A accepts L, so H must be ultimately increasing, in the sense that there exists an infinite set $\mathbb{I} \subseteq \mathbb{N}$ such that for all $i \in \mathbb{I}$ and $j > i$ it holds that $H(j) > H(i)$.

Since $\mathbb{I}$ is infinite and $Q \times \Gamma$ finite there must exist some (q, γ) for which $\mathbb{I}_{(q,\gamma)} = \{i \in \mathbb{I} \mid S(i) = (q, \gamma)\}$ is infinite. Let $n = |Q| + 1$ and let $i_1, \ldots, i_n$ be the n smallest elements of $\mathbb{I}_{(q,\gamma)}$.

To give the intuition of the remainder of the proof, we now select a sufficiently long prefix of ω that all of the above configurations are visited. A suffix which is in the language is selected, then a substring of the prefix is found such that pumping it will force an equivalent pumping in the suffix.

Pick any string v such that $\omega(i_n) \cdot v \in \mathcal{L}$ (at least one must exist as $(\omega(i) \propto L) \neq \emptyset$ for all i). Let $w = \omega(i_n) \cdot v$ and let S', H' be functions as above but for running A on w. Let $j_1, \ldots, j_n \in \mathbb{N}$ be such that, for all k, j_k is the smallest number with $j_k > i_k$ and $H'(j_k) \leq H'(i_k)$. More precisely this means that $H'(j_k) = H'(i_k)$, and the configurations reached by A on $w(j_k)$ and $w(i_k)$ have the same stack. This is necessarily the case since A is stack-emptying and a rule can pop at most one stack symbol. Notice

that the j indices end up being "reversed", in that $j_{k+1} < j_k$ for all k. Since $n > |Q|$ the pigeon-hole principle yields that there exists some $p \in Q$ and $k_1, k_2 \in \mathbb{N}$ (with $k_1 < k_2$) such that $(p, \gamma) = S(j_{k_1}) = S(j_{k_2})$. Note that this is the same γ as in $S(i_1)$ through $S(i_n)$.

Next, we partition w into $w = xvyuz$ where

- $x = w(1, i_{k_1})$,
- $v = w(i_{k_1} + 1, i_{k_2})$,
- $y = w(i_{k_2} + 1, j_{k_2})$,
- $u = w(j_{k_2} + 1, j_{k_1})$,
- $z = w(j_{k_1} + 1, |w|)$.

Notice that after A has read x it will be in a configuration of the form $(q, \gamma s)$, and that, by construction, whenever A is in a configuration of the form $(q, \gamma s')$ it can read v to go to a configuration $(q, \gamma s'' \gamma s')$ without ever inspecting s'. In the opposite direction, whenever A is in a configuration of the form $(p, \gamma s'' \gamma s')$ (with the same s' as above) it can read u to go to the configuration $(p, \gamma s')$, without ever inspecting s'. In particular, since $xvyuz \in L$ we have $xv^i yu^i z \in L$ for all $i \geq 1$.

Claim. $xv^i \not\equiv_L xv^j$ for all $1 \leq i < j$.

Assume that there are $1 \leq i < j$ such that $xv^i \equiv_L xv^j$. Let $d = j - i$. Then, since equivalence is closed under concatenation to the right, $xv^i \equiv_L xv^{i+d} \equiv_L xv^{i+2d} \equiv_L xv^{i+3d} \equiv_L \cdots$. However, as argued above, running A on the string xv^{i+cd}, for $c \in \mathbb{N}$, will place it into a configuration

$$(q, \underbrace{\gamma s'' \gamma s'' \cdots \gamma s''}_{i+cd \text{ copies}} \gamma s),$$

which, for a sufficiently large c will give a stack deeper than $|yu^i z|$, but since A must empty its stack before accepting, and must read a symbol for each symbol popped from the stack, this means that $xv^{i+cd} yu^i z \notin L$, but $xv^i yu^i z \in L$, so our assumption that $xv^j \equiv_L xv^i$ must have been incorrect. This contradiction proves the claim, which makes x and v fulfill the conjecture for L. ∎

It seems probable that this proof can be extended to handle slightly less restrictive language classes (notably the stack emptying is a candidate for removal), but introducing full non-determinism appears to require a more advanced approach.

6 Closure Properties of Shuffling Context-Free Languages

To show that Conjecture 4 is not entirely without motivation (although it appears very interesting in its own right), we will here prove a fairly interesting statement about the shuffle of context-free languages that can be made using the conjecture. First let us recall the definition of the *shuffle*, an interleaving operator of great interest in many areas.

Definition 8 For all strings $w = \alpha_1 \cdots \alpha_n$ and $v = \beta_1 \ldots \beta_m$, for $\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_m \in \Sigma$, let $w \odot v$ denote the *shuffle* of w and v. The shuffle is defined by letting $w \odot \varepsilon = \varepsilon \odot w = w$, and letting

$$\alpha_1 \cdots \alpha_n \odot \beta_1 \cdots \beta_m = \alpha_1(\alpha_2 \cdots \alpha_n \odot \beta_1 \cdots \beta_m) \cup \beta_1(\alpha_1 \cdots \alpha_n \odot \beta_2 \cdots \beta_m).$$

For languages $L, L' \subseteq \Sigma^*$ let $L \odot L' = \{w \odot v \mid w \in L, v \in L'\}$. ◇

Next, let us for the sake of completeness recall the statement of Ogden's Lemma [Ogd68], which generalizes the pumping lemma of Definition 3, and which will be useful later.

Lemma 9 If a language $\mathcal{L} \in \Sigma^*$ is context-free there exists a constant $p \in \mathbb{N}$ such that for every string $w \in \mathcal{L}$ and every way of marking at least p of the positions in w there exists a subdivision $p_1 r_1 m r_2 p_2 = w$ such that

1. $r_1 r_2$ contains at least one marked position,
2. $r_1 m r_2$ contains at most p marked positions,
3. $p_1 r_1^i m r_2^i p_2 \in \mathcal{L}$ for all $i \in \{0, 1, \ldots\}$. ◇

We can now go on to prove the following theorem.

Theorem 10 Assume that Conjecture 4 holds. Let a, b be two symbols not in Σ. Let $\mathcal{D} = \{a^n b^n \mid n \in \mathbb{N}\}$. Then for any *context-free* $\mathcal{L} \in \Sigma^*$ it holds that $\mathcal{L} \odot \mathcal{D} \in \mathcal{CF}$ if and only if $L \in \mathcal{Reg}$. ◇

PROOF: If $\mathcal{L} \in \mathcal{Reg}$ it is trivially true that $\mathcal{L} \odot \mathcal{D} \in \mathcal{CF}$, one can directly construct a push-down automaton which recognizes the regular $\mathcal{L}$ entirely in its finite state, and uses the stack to recognize the $\mathcal{D}$ portion.

The remainder of the proof demonstrates the other direction. Using $\mathcal{L}$ let $x, v \in \Sigma^*$ be as in Conjecture 4, which we assumed to be true. Let $\mathcal{L}' = \mathcal{L} \odot \mathcal{D}$ as above. The proof continues by contradiction, let us assume that $\mathcal{L}' \in \mathcal{CF}$ but $\mathcal{L} \in \mathcal{CF} \setminus \mathcal{Reg}$.

Let $\mathcal{R}$ be the regular language corresponding to the regular expression $x(av)^* b^* \Sigma^*$, then let $\mathcal{L}'' = \mathcal{L}' \cap \mathcal{R}$. This $\mathcal{L}''$ must be context-free, since $\mathcal{L}'$ is assumed to be context-free and $\mathcal{CF}$ is closed under intersection with regular languages.

Let p be the constant for which (Ogden's) Lemma 9 holds for $\mathcal{L}''$. For all $\gamma \in \mathbb{N}$ let $W_\gamma = xv^\gamma \propto \mathcal{L}$. Then, for each $\gamma \geq p$ and $w \in W_\gamma$ consider the string $x(av)^\gamma b^\gamma w$, which is in $\mathcal{L}''$ by definition. Apply Lemma 9 by marking all symbols in the substring $(av)^\gamma$ (that is sufficient, as we chose $\gamma \geq p$), and let $p_1 r_1 m r_2 p_2$ be the subdivision that the lemma dictates exists. We know that $r_1 r_2$ must contain at least one symbol from $(av)^\gamma$, and that $p_1 r_1^i m r_2^i p_2 \in \mathcal{L}''$ for all $i \geq 0$ by the lemma, this restricts the choice of r_1 and r_2 severely:

- The language $\mathcal{R}$ used in the construction of $\mathcal{L}''$ ensures that if any symbol in $(av)^\gamma$ is repeated then a complete substring of the form $(av)^k$, for $1 \leq k \leq p$ must be repeated. In actuality $k \leq \left\lfloor \frac{p}{|av|} \right\rfloor$ is dictated by Lemma 9, but we overestimate for simplicity.
- The intersection with the language $\mathcal{R}$ also ensures that neither r_1 nor r_2 can span the border between x and $(av)^\gamma$, the border between $(av)^\gamma$ and b^γ, or the border between b^γ and w, as those borders are distinctly dictated in the regular expression.

- If r_1 contains no a then r_2 must contain the a, but then the repetition $p_1 r_1^i m r_2^i p_2$ increases the number of as without increasing the number of bs, which the shuffled in language $\mathcal{D}$ disallows. As such, r_1 must contain an a, and therefore be of the form $(av)^k$ for some k, as this is the only repetition in this section that $\mathcal{R}$ makes possible.
- This leaves r_2 to be of the form b^k, to preserve the number of bs in correspondence with the number of as.

This leads us to w, which by the above cannot be modified by the pumping $p_1 r_1^i m r_2^i p_2$ (i.e., it necessarily falls entirely within p_2), despite the number of vs changing as part of the pumping of a substring in $(av)^*$. This means that for all $\gamma \geq p$ and $w \in W_\gamma$ (recall, $W_\gamma = xv^\gamma \propto \mathcal{L}$) there exists a constant k such that $xv^{\gamma+ik}w \in \mathcal{L}$ for all $i \geq -1$.

Notice that $p!$ is a multiple of all $1 \leq k \leq p$, meaning that all $w \in W_\gamma$ are such that $xv^{\gamma+i(p!)}w \in \mathcal{L}$ for all $i \geq 0$. It follows from this that $W_\gamma \subseteq W_{\gamma+p!}$, and, since Conjecture 4 is assumed to be true $W_i \neq W_j$ for all $i \neq j$, so the inclusion is strict: $W_\gamma \subsetneq W_{\gamma+p!}$ for all $\gamma \geq p$. Since this holds for all γ we can by simple induction establish that

$$W_p \subsetneq W_{p+p!} \subsetneq W_{p+2(p!)} \subsetneq W_{p+3(p!)} \subsetneq \cdots$$

by simply choosing γ to be p, $p+p!$, $p+2(p!)$ successively. Replacing the uses of W_γ with its left quotient definition we arrive at the statement that

$$(xv^{p+i(p!)} \propto \mathcal{L}) \subsetneq (xv^{p+(i+1)(p!)} \propto \mathcal{L})$$

for all $i \geq 0$. Now, for each $i > 1$ pick some representative $w_i \in (xv^{p+i(p!)} \propto \mathcal{L}) \setminus (xv^{p+(i-1)(p!)} \propto \mathcal{L})$. Notice that this by induction means that $w_i \notin (xv^{p+j(p!)} \propto \mathcal{L})$ for any $0 < j < i$.

Next, let $\hat{\mathcal{R}} = xv^p(av^{p!})^* b^* \Sigma^*$, and construct $\hat{\mathcal{L}} = \hat{\mathcal{R}} \cap (\mathcal{L} \odot \mathcal{D})$. Let $\hat{p}$ be the constant for Lemma 9 for this language. Pick the string $xv^p(av^{p!})^{\hat{p}} b^{\hat{p}} w_{\hat{p}}$. Apply Lemma 9 by marking the substring $(av^{p!})^{\hat{p}}$ and let $p_1 r_1 m r_2 p_2$ be the substring subdivision that the lemma dictates exist. Again notice that r_1 must fall entirely within the substring $(av^{p!})^{\hat{p}}$ due to the intersection with $\hat{\mathcal{R}}$, and r_2 must then fall entirely within the b^* substring (refer to the full argument above). The lemma then further dictates that $p_1 m p_2 \in \hat{\mathcal{L}}$, but this removes (using the same argument for where r_1 and r_2 must be in the original string as above) a non-zero number of the substrings $av^{p!}$ from the string, and by construction $w_{\hat{p}} \notin (xv^{p+k(p!)} \propto \mathcal{L})$ for $k < \hat{p}$.

This contradicts the assumption that $\mathcal{L} \in \mathcal{CF}$ and $\mathcal{L}' \notin \mathcal{Reg}$. ∎

7 Conclusions

We have shown that Conjecture 4 holds for at least one limited case, and proven a follow-up result of an interesting nature. However, the most obvious missing piece is a complete proof of the conjecture, or possibly a counter-example. The author still hopes to manage such a proof, but contributions are most certainly welcome. Secondarily an extension of the proof that the shuffle of a context-free language with $a^n b^n$ (on disjoint alphabets) is context-free if and only if the first language was regular should also be

considered. It appears likely that for all $\mathcal{L}, \mathcal{L}' \in \mathcal{CF}$, on disjoint alphabets, $\mathcal{L} \odot \mathcal{L}' \in \mathcal{CF}$ if and only if either $\mathcal{L} \in \mathcal{Reg}$ or $\mathcal{L}' \in \mathcal{Reg}$. In this case it seems likely that this can be achieved along the lines of the proof of Theorem 10, by making the central claims symmetrical and employing some case analysis, but a counter-example as the final result cannot yet be entirely ruled out.

References

[BHPS61] Yehoshua Bar-Hillel, Micha A. Perles, and Eli Shamir. On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*, (14):143–172, 1961.

[CS63] Noam Chomsky and Marcel Paul Schützenberger. The Algebraic Theory of Context-Free Languages. In P. Braffort and D. Hirshberg, editors, *Computer Programming and Formal Systems*, Studies in Logic, pages 118–161. North-Holland Publishing, Amsterdam, 1963.

[HMU03] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation (2nd Ed.)*. Pearson Education International, 2003.

[Ogd68] William Ogden. A helpful result for proving inherent ambiguity. *Mathematical systems theory*, 2(3):191–194, 1968.

[Par66] Rohit Parikh. On context-free languages. *J. ACM*, 13(4):570–581, 1966.