Shuffling and Unshuffling

Dane Henshall
School of Computer Science
University of Waterloo
Waterloo, ON N2L 3G1
Canada
dslhensh@uwaterloo.ca

Narad Rampersad
Department of Mathematics
University of Liège
Grande Traverse, 12 (Bat. B37)
4000 Liège
Belgium
narad.rampersad@gmail.com

Jeffrey Shallit
School of Computer Science
University of Waterloo
Waterloo, ON N2L 3G1
Canada
shallit@cs.uwaterloo.ca

Abstract

We consider various shuffling and unshuffling operations on languages and words, and examine their closure properties. Although the main goal is to provide some good and novel exercises and examples for undergraduate formal language theory classes, we also provide some new results and some open problems.

1 Introduction

Two kinds of shuffles are commonly studied: perfect shuffle and ordinary shuffle.

For two words $x = a_1 a_2 \cdots a_n$, $y = b_1 b_2 \cdots b_n$ of the same length, we define their *perfect shuffle* $x \mathbin{\text{ш}} y = a_1 b_1 a_2 b_2 \cdots a_n b_n$. For example, $\texttt{term} \mathbin{\text{ш}} \texttt{hoes} = \texttt{theorems}$. Note that $x \mathbin{\text{ш}} y$ need not equal $y \mathbin{\text{ш}} x$. This definition is extended to languages as follows:

$$L_1 \mathbin{\text{ш}} L_2 = \bigcup_{\substack{x \in L_1,\ y \in L_2 \\ |x|=|y|}} \{x \mathbin{\text{ш}} y\}.$$

If x^R denotes the reverse of x, then note that $(x \mathbin{\text{ш}} y)^R = y^R \mathbin{\text{ш}} x^R$.

It is sometimes useful to allow $|y| = |x| + 1$, where $x = a_1 \cdots a_n$, $y = b_1 \cdots b_{n+1}$, in which case we define $x \mathbin{\text{ш}} y = a_1 b_1 \cdots a_n b_n b_{n+1}$.

The *ordinary shuffle* $x \mathbin{\text{Ш}} y$ of two words is a finite *set*, the set of words obtainable from merging the words x and y from left to right, but choosing the next symbol arbitrarily from x or y. More formally,

$$\begin{aligned} x \mathbin{\text{Ш}} y = \{z \ : \ & z = x_1 y_1 x_2 y_2 \cdots x_n y_n \text{ for some } n \geq 1 \text{ and} \\ & \text{words } x_1, \ldots, x_n, y_1, \ldots, y_n \text{ such that } x = x_1 \cdots x_n \text{ and } y = y_1 \cdots y_n\}. \end{aligned}$$

This definition is symmetric, and $x \mathbin{\text{Ш}} y = y \mathbin{\text{Ш}} x$. The definition is extended to languages as follows:

$$L_1 \mathbin{\text{Ш}} L_2 = \bigcup_{x \in L_1,\ y \in L_2} (x \mathbin{\text{Ш}} y).$$

(As a mnemonic, the symbol Ш is larger than ш in size, and similarly Ш generally produces a set larger in cardinality than ш.)

As is well-known, the shuffle (resp., perfect shuffle) of two regular languages is regular, and the shuffle (resp., perfect shuffle) of a context-free language with a regular language is context-free. Perhaps the easiest way to see all these results is by using morphisms and inverse morphisms, and relying on the known closure properties of these transformations, as follows:

If $L_1, L_2 \subseteq \Sigma^*$, create a new alphabet Σ' by putting primes on all the letters of Σ. Define $h_1(a) = h_2(a') = a$ and $h_1(a') = h_2(a) = \epsilon$ for $a \in \Sigma$. Define $h(a) = h(a') = a$ for $a \in \Sigma$. Then

$$L_1 \mathbin{\text{Ш}} L_2 = h(h_1^{-1}(L_1) \ \cap \ h_2^{-1}(L_2)).$$

In a similar way,

$$L_1 \mathbin{\text{ш}} L_2 = h(h_1^{-1}(L_1) \ \cap \ h_2^{-1}(L_2) \ \cap (\Sigma\Sigma')^*).$$

However, the shuffle (resp., perfect shuffle) of two context-free languages need not be context-free. For example, if $L_1 = \{a^m b^m \ : \ m \geq 1\}$ and $L_2 = \{c^n d^n \ : \ n \geq 1\}$, then $L := L_1 \mathbin{\text{Ш}} L_2$ is not a CFL. If it were, then $L \ \cap \ a^+ c^+ b^+ d^+ = \{a^m c^n b^m d^n \ : m, n \geq 1\}$ would be a CFL, which it isn't (via the pumping lemma).

Similarly, if $L_3 = \{a^m b^{2m} \ : \ m \geq 1\}$ and $L_4 = \{a^{2n} b^n \ : \ n \geq 1\}$, then $L_3 \mathbin{\text{ш}} L_4 = \{a^{2n}(ba)^n b^{2n} \ : \ n \geq 1\}$, which is clearly not a CFL.

For these, and other facts, see [1].

2 Self-shuffles

Instead of shuffling languages together, we can take a language and shuffle (resp., perfect shuffle) each word with itself. Another variation is to shuffle each word with its reverse. This gives four different transformations on languages, which we call self-shuffles:

$$
\begin{aligned}
\mathrm{ss}(L) &= \bigcup_{x \in L} \{x \mathbin{\text{Ш}} x\} \\
\mathrm{pss}(L) &= \bigcup_{x \in L} x \mathbin{\text{ш}} x \\
\mathrm{ssr}(L) &= \bigcup_{x \in L} \{x \mathbin{\text{Ш}} x^R\} \\
\mathrm{pssr}(L) &= \bigcup_{x \in L} x \mathbin{\text{ш}} x^R.
\end{aligned}
$$

We would like to understand how these transformations affect regular and context-free languages. We obtain some results, but other questions are still open.

Theorem 1. *If L is regular, then* $\mathrm{ss}(L)$ *need not be context-free.*

Proof. We show that $\mathrm{ss}(\{0,1\}^*)$ is not a CFL. Suppose it is, and consider $L' = \mathrm{ss}(\{0,1\}^*) \cap R$, where $R = \{01^a 0^{b+1} 1^{c+1} 0^d 1 \ : \ a, b, c, d \geq 1\}$. Since R is regular, it suffices to show that L' is not context-free.

Now consider an arbitrary word $w \in L'$. Then $w = 01^a 0^{b+1} 1^{c+1} 0^d 1$ for some $a, b, c, d \geq 1$, and there exists a $y \in \{0,1\}^*$ such that $w \in y \mathbin{\text{Ш}} y$. The structure of w allows us to determine y. Let y_1 and y_2 be copies of y such that $w \in y_1 \mathbin{\text{Ш}} y_2$, and the first letter of w is taken from y_1.

The first symbol of y is evidently 0. It follows that the prefix 01^a of w is taken entirely from y_1, since the 0 is taken from y_1 by definition and the first symbol of y_2 is 0. Therefore 01^a is a prefix of y_1.

It follows that y_2 also contains 01^a as a prefix, and since $a \geq 1$ this is only possible if the first 0 of y_2 is located in the 0^{b+1} block of w. Otherwise, y_2 would be a subsequence of $0^d 1$ and y_1 would have $01^a 0^{b+1} 1^{c+1}$ as a prefix (implying that $y_1 \neq y_2$). Furthermore, the second symbol of y_2 being 1 implies that exactly one of the 0's in the 0^{b+1} block is from y_2. Thus the rest are from y_1 and $01^a 0^b$ is a prefix of y_1.

Note that y_1 and y_2 both end in 1, and w ends in 0^d1. By the same logic as before, we can conclude that 0^d1 is a suffix of exactly one of them, and that the other ends in the 1^{c+1} block. Thus y_2 contains 0^d1 as a suffix and y_1 ends in the 1^{c+1} block (otherwise, $y_1 \neq y_2$).

Finally, since the second last symbol of y_1 is 0 and y_1 ends in the 1^{c+1} block, we can conclude that y_1 contains exactly one 1 from the 1^{c+1} block and that $y_1 = 01^a0^b1$. Unshuffling y_1 from w yields $y_2 = 01^c0^d1$.

Recall that $y_1 = y_2$. So,

$$y_1 = 01^a0^b1 = 01^c0^d1 = y_2$$

and since $a, b, c, d \geq 1$ we know that

$$a = c \qquad \text{and} \qquad b = d.$$

If $w \in L'$ then

$$\begin{aligned} w &= 01^a0^{b+1}1^{c+1}0^d1 \\ &= 01^a0^{d+1}1^{a+1}0^d1 \\ &= 01^a0^d(01)1^a0^d1. \end{aligned}$$

Since w was arbitrary, we have

$$\begin{aligned} L' &= \{01^a0^{b+1}1^{c+1}0^d1 : a = c, b = d, \text{ and } a, d \geq 1\} \\ &= \{01^n0^m(01)1^n0^m1 : m, n \geq 1\}, \end{aligned}$$

which is clearly not a CFL, using the pumping lemma. □

Remark 2. In a previous version of this paper, proving that $\mathrm{ss}(\{0,1\}^*)$ is not context-free was listed as an open problem. After this was solved by D. Henshall, a solution was given by Georg Zetzsche independently.

Similarly, we can show

Theorem 3. $L = \bigcup_{w \in \{0,1\}^*} (w \sqcup\!\sqcup\, w \sqcup\!\sqcup\, w)$ *is not context-free.*

Proof. We use Ogden's lemma. Consider

$$L = \{w \sqcup\!\sqcup\, w \sqcup\!\sqcup\, w \ : \ w \in \{0,1\}^*\} \cap 0^*10^*10^*1.$$

Pick $s = 0^n10^n10^n1$ in L to pump. Write $s = uvxyz$ and mark the middle block of 0's. If v begins in the middle block of 0's, then pump up to obtain $s' = 0^n10^j10^k1$, where $n < j$ and $n \leq k$. We can't have $s' \in w \sqcup\!\sqcup\, w \sqcup\!\sqcup\, w$ because the first w (the one ending at the first 1) is too short. If v begins in the first block of 0's, then y occurs in the middle block, so now pump down to obtain $s' = 0^i 1 0^j 1 0^n 1$, where $i \leq n$ and $j < n$. Again, we can't have $s' \in w \text{ Ш } w \text{ Ш } w$, because the third w (the one ending at the third 1) must contain all of the 0's immediately preceding the final 1, and hence is too long. $\square$

Clearly $\mathrm{ss}(\{0,1\}^*)$ is in NP, since given a word w we can guess x and check that $w \in x \text{ Ш } x$. However, we do not know whether we can solve membership for $\mathrm{ss}(\{0,1\}^*)$ in polynomial time. This question is apparently originally due to Jeff Erickson [2], and we learned about it from Erik Demaine.

Open Problem 4. Is $\mathrm{ss}(\{0,1\}^*)$ in P?

We mention a few related problems. Mansfield [4] showed that, given words w, x, y, one can decide in polynomial time if $w \in x \text{ Ш } y$. Later, the same author [5] and, independently, Warmuth and Haussler [6] showed that, given words $w, x_1, x_2, \ldots, x_n$, deciding if $w \in x_1 \text{ Ш } x_2 \text{ Ш } \cdots \text{ Ш } x_n$ is NP-complete. However, the decision problem implied by Open Problem 4 asks something different: given w, does there exist x such that $w \in x \text{ Ш } x$?

Open Problem 5. Determine a simple closed form for

$$a_k(n) := \left| \bigcup_{x \in \{0,1,\ldots,k-1\}^n} (x \text{ Ш } x) \right|.$$

The first few terms are given as follows:

n	0	1	2	3	4	5	6	7	8	9
$a_2(n)$	1	2	6	22	82	320	1268	5102	20632	83972
$a_3(n)$	1	3	15	93	621	4425	32703	248901		
$a_4(n)$	1	4	28	244	2332	23848	254416			
$a_5(n)$	1	5	45	505	6265	83225				
$a_6(n)$	1	6	66	906	13806	225336				

Clearly $a_i(0) = 1$, $a_i(1) = i$, and $a_i(2) = 2i^2 - i$. Empirically we have $a_i(3) = 5i^3 - 5i^2 + i$, $a_i(4) = 14i^4 - 21i^3 + 5i^2 + 3i$, and $a_i(5) = 42i^5 - 84i^4 + 32i^3 + 21i^2 - 10i$. This suggests that $a_i(n) = \frac{\binom{2n}{n}}{n+1} i^n - \binom{2n-1}{n+1} i^{n-1} + O(i^{n-2})$, but we do not have a proof.

3 Perfect self-shuffle

We can consider the same question for perfect shuffle. We define

$$\mathrm{pss}(L) = \bigcup_{x \in L} \{x \amalg x\}.$$

Theorem 6. *Both the class of regular languages and the class of context-free languages are closed under* pss*.*

Proof. Use the fact that $\mathrm{pss}(L) = h(L)$, where h is the morphism mapping $a \to aa$ for each letter a. □

4 Self-shuffle with reverse

We now characterize those words y that can be written as a shuffle of a word with its reverse; that is, as a member of the set $x \amalg x^R$.

An *abelian square* is a word of the form xx' where x' is a permutation of x.

Theorem 7. *(a) If there exists x such that $y \in x \amalg x^R$, then y is an abelian square.*

(b) If y is a binary abelian square, then there exists x such that $y \in x \amalg x^R$.

We introduce the following notation: if $w = a_1 a_2 \cdots a_n$, then by $w[i..j]$ we mean the factor $a_i a_{i+1} \cdots a_j$.

Proof. (a) If y is the shuffle of x with its reverse, then the first half of y must contain some prefix of x, say $x[1..k]$. Then the second half of y must contain the remaining suffix of x, say $x[k + 1..n]$. Then the second half of y must contain, in the remaining positions, some prefix of x, reversed. But by counting we see that this prefix must be $x[1..k]$. So the first half of y must contain the remaining symbols of x, reversed. This shows that the first half of y is just $x[1..k]$ shuffled with $x[k+1..n]^R$, and the second half of y is just $x[k+1..n]$ shuffled with $x[1..k]^R$.

So the second half of y is a permutation of the first half of y.

(b) It remains to see that every binary abelian square can be obtained in this way.

To see this, note that if x contains j 0's and $n - j$ 1's, then we can get y by shuffling $0^j 1^{n-j}$ with its reverse. We get the 0's in x by choosing them from $0^j 1^{n-j}$, and we get the 1's in x by choosing them from $(0^j 1^{n-j})^R$. □

Remark 8. The word 012012 is an example of a ternary abelian square that cannot be written as an element of $w \amalg w^R$ for any word w.

Remark 9. The preceding proof gives another proof of the classic identity

$$\binom{2n}{n} = \binom{n}{0}^2 + \cdots + \binom{n}{n}^2.$$

To see this, we use the following bijections: the binary words of length $2n$ having exactly n 0's (and hence n 1's) are in one-one correspondence with the abelian squares of length $2n$, as follows: take such a word and complement the last n bits. Thus there are $\binom{2n}{n}$ binary abelian squares of length $2n$.

On the other hand, there are $\binom{n}{i}^2$ words that are abelian squares and have a first and last half, each with i 0's. Summing this from $i = 0$ to n gives the result.

Corollary 10. *The language*

$$\mathrm{ssr}(\{0,1\}^*) = \bigcup_{x \in \{0,1\}^*} x ⧢ x^R$$

is not a CFL, but is in P.

Proof. From above, intersecting $\mathrm{ssr}(\{0,1\}^*)$ with $0^+1^+0^+1^+$ gives

$$\{0^m 1^n 0^{m+2k} 1^n \ : \ m, n \geq 1 \text{ and } k \geq 0\} \cup \{0^m 1^{n+2k} 0^m 1^n \ : \ m, n \geq 1 \text{ and } k \geq 0\}.$$

Now the pumping lemma applied to $z = 0^n 1^n 0^n 1^n$ shows this is not a CFL.

Since we can easily test if a string is an abelian square by counting the number of 0's in the first half, and comparing it to the number of 0's in the second half, it follows that $\mathrm{ssr}(\{0,1\}^*)$ is in P. □

As before, we can define

$$b_k(n) := \left| \bigcup_{x \in \{0,1,\ldots,k-1\}^n} (x ⧢ x^R) \right|.$$

For $k = 2$, our results above explain $b_k(n)$, but we do not know a closed form for larger k.

The first few terms are given as follows:

n	0	1	2	3	4	5	6	7	8	9
$b_2(n)$	1	2	6	20	70	252	924	3432	12870	48620
$b_3(n)$	1	3	15	87	549	3657	25317	180459		
$b_4(n)$	1	4	28	232	2116	20560	208912			
$b_5(n)$	1	5	45	485	5785	73785				
$b_6(n)$	1	6	66	876	12906	203676				

Clearly $b_i(0) = 1$, $b_i(1) = i$, and $b_i(2) = 2i^2 - i$. Empirically, we have $b_i(3) = 5i^3 - 6i^2 + 2i$, $b_i(4) = 14i^4 - 27i^3 + 17i^2 - 3i$, and $b_i(5) = 42i^5 - 110i^4 + 94i^3 - 17i^2 - 8i$. This suggests that $b_i(n) = \frac{\binom{2n}{n}}{n+1} i^n - \left(\binom{2n-1}{n-1} - 2^{n-1}\right) i^{n-1} + O(i^{n-2})$, but we do not have a proof.

5 Perfect self-shuffle with reverse

We now consider the operation $w \rightarrow w \text{ ш } w^R$ applied to languages. Recall that $\mathrm{pssr}(L) = \bigcup_{x \in L} \{x \text{ ш } x^R\}$.

Theorem 11. *If L is regular then* pssr(L) *is not necessarily regular.*

Proof. Let $L = 0^+10^+$. Then $\mathrm{pssr}(L) \cap 0^+110^+ = \{0^n110^n \ : \ n \geq 2\}$, which is clearly not regular. □

Theorem 12. *If L is context-free then* pssr(L) *is not necessarily context-free.*

Proof. Let $L = \{0^m1^m2^n3^n \ : \ m, n \geq 1\}$. Then $\mathrm{pssr}(L) \cap (03)^+(12)^+(21)^+(30)^+ = \{(03)^n(12)^n(21)^n(30)^n \ : \ n \geq 1\}$, and this language is easily seen to be non-context-free. □

Theorem 13. *If L is regular then* pssr(L) *is necessarily context-free.*

We defer the proof of Theorem 13 until Section 6.4 below.

6 Unshuffling

Given a finite word $w = a_1a_2 \cdots a_n$ we can decimate it into its odd- and even-indexed parts, as follows:

$$\begin{aligned} \mathrm{odd}(w) &= a_1a_3 \cdots a_{n-((n+1) \bmod 2)} \\ \mathrm{even}(w) &= a_2a_4 \cdots a_{n-(n \bmod 2)} \end{aligned}$$

Similarly, given $w = a_1a_2 \cdots a_n$ we can extract its first and last halves, as follows:

$$\begin{aligned} \mathrm{fh}(w) &= a_1a_2 \cdots a_{\lfloor n/2 \rfloor} \\ \mathrm{lh}(w) &= a_{\lfloor n/2 \rfloor + 1} \cdots a_n \end{aligned}$$

We now turn our attention to four "unshuffling" operations:

$$\begin{aligned} \mathrm{bd}(w) &= \mathrm{odd}(w)\mathrm{even}(w) \\ \mathrm{bdr}(w) &= \mathrm{odd}(w)\mathrm{even}(w)^R \\ \mathrm{bdi}(w) &= \mathrm{fh}(w) \text{ ш } \mathrm{lh}(w) \\ \mathrm{bdir}(w) &= \mathrm{fh}(w) \text{ ш } \mathrm{lh}(w)^R \end{aligned}$$

6.1 Binary decimation

We first consider a kind of binary decimation, which forms a sort of inverse to perfect shuffle.

Given a word $w = a_1 a_2 \cdots a_{2n}$ of even length, note that

$$\mathrm{bd}(w) = a_1 a_3 \cdots a_{2n-1} a_2 a_4 \cdots a_{2n}$$

is formed by "unshuffling" the word into its odd- and even-indexed letters. For example, the French word `maigre` becomes the word `mirage` under this operation.

Theorem 14. *Neither the class of regular languages nor the class of context-free languages is closed under* bd.

Proof. Consider the regular (and context-free) language $L = (00 + 11)^+$. Then $\mathrm{bd}(L) = \{ww \ : \ w \in \{0,1\}^+\}$, which is well-known to be non-context-free. ◻

6.2 Binary decimation with reverse

We now consider the operation bdr, which is a kind of binary decimation with reverse. Note that

$$\mathrm{bdr}(a_1 a_2 \cdots a_{2n}) = a_1 a_3 \cdots a_{2n-1} a_{2n} \cdots a_4 a_2.$$

For example, bdr(`friend`) = `finder` and bdr(`perverse`) = `preserve`.

Theorem 15. *The class of regular languages is not closed under* bdr.

Proof. Let $L = (00)^+ 11$. Then $\mathrm{bdr}(L) = \{0^n 110^n \ : \ n \geq 1\}$, which is not regular. ◻

Theorem 16. *The class of context-free languages is not closed under* bdr.

Proof. Consider $L = \{(03)^n (12)^n \ : \ n \geq 1\}$. Then $\mathrm{bdr}(L) = \{0^n 1^n 2^n 3^n \ : \ n \geq 1\}$, which is not context-free. ◻

Theorem 17. *If L is regular, then* $\mathrm{bdr}(L)$ *is context-free.*

Proof. We show how to accept words of $\mathrm{bdr}(L)$ of even length; words of odd length can be treated similarly.

On input $w = b_1 b_2 \cdots b_{2n}$, a PDA can guess $x = a_1 a_2 \cdots a_{2n}$ in parallel with the elements of the input. At each stage the PDA compares a_i to $b_{(i+1)/2}$ if i is odd; and otherwise it pushes a_i onto the stack (if i is even). At some point the PDA nondeterministically guesses that it has seen a_{2n} and pushed it on the stack; it now pops the stack (which is holding $a_{2n} \cdots a_4 a_2$) and compares the stack contents to the rest of the input w.

The PDA accepts if $x \in L$ and the symbols matched as described. ◻

6.3 Inverse decimation

We now consider a kind of inverse decimation, which shuffles the first and last halves of a word.

Note that if $w = a_1 \cdots a_{2n}$ is of even length, then

$$\mathrm{bdi}(w) = a_1 a_{n+1} a_2 a_{n+2} \cdots a_n a_{2n}.$$

Further, $\mathrm{bdi}(\mathrm{bd}(w)) = \mathrm{bd}(\mathrm{bdi}(w))$ for w of even length.

Theorem 18. *If L is regular then so is* $\mathrm{bdi}(L)$*.*

Proof. On input x we simulate the DFA for L on the odd-indexed letters of x, starting from q_0, and we simulate a second copy of the DFA for L on the even-indexed letters, starting at some guessed state q. Finally, we check to see that our guess of q was correct. □

Theorem 19. *The class of context-free languages is not closed under* bdi*.*

Proof. Let $L = \{0^m 1^m 2^{2n} 3^{4n} \ : \ m, n \geq 1\}$. It is easy to see that

$$\mathrm{bdi}(L) = \begin{cases} (01)^{m-3n}(02)^{2n}(03)^n(13)^{3n}, & \text{if } m \geq 3n; \\ (02)^{m-n}(03)^n(13)^m(23)^{3n-m}, & \text{if } n \leq m \leq 3n; \\ (03)^m(13)^m(23)^{2n}(33)^{n-m}, & \text{if } m \leq n. \end{cases}$$

Consider $L' := \mathrm{bdi}(L) \ \cap \ (03)^+(13)^+(23)^+$. From the above we have $L' = \{(03)^n(13)^n(23)^{2n} \ : \ n \geq 1\}$, which is evidently not context-free. □

6.4 Inverse decimation with reverse

Note that if $w = a_1 \cdots a_{2n}$ is of even length, then $\mathrm{bdir}(w) = a_1 a_{2n} a_2 a_{2n-1} \cdots a_n a_{n+1}$. If $w = a_1 \cdots a_{2n+1}$ is of odd length, we define

$$\mathrm{bdir}(w) = a_1 a_{2n+1} a_2 a_{2n} \cdots a_n a_{n+2} a_{n+1}.$$

Theorem 20. *If L is regular then so is* $\mathrm{bdir}(L)$*.*

Proof. On input x we simulate the DFA M for L on the odd-indexed letters of x, starting from q_0. We also create an NFA M' accepting L^R in the usual manner, by reversing the transitions of M, and making the start state the set of final states of M, and we simulate M' on the even-indexed letters of x. Finally, we check to see that we meet in the middle. □

Theorem 21. *The class of context-free languages is not closed under* bdir*.*

Proof. Consider $L = \{0^{2m}1^{4m}2^n3^n \ : \ m, n \geq 1\}$. Then L is a CFL, and it is easy to verify that

$$\mathrm{bdir}(0^{2m}1^{4m}2^n3^n) = \begin{cases} (03)^n(02)^n(01)^{2m-2n}(11)^{m+n}, \text{if } m \geq n; \\ (03)^n(02)^{2m-n}(12)^{2n-2m}(11)^{3m-n}, \text{if } m \leq n \leq 2m; \\ (03)^{2m}(13)^{n-2m}(12)^n(11)^{3m-n}, \text{if } 2m \leq n \leq 3m; \\ (03)^{2m}(13)^{n-2m}(12)^{6m-n}(22)^{n-3m}, \text{if } 3m \leq n \leq 6m; \\ (03)^{2m}(13)^{4m}(23)^{n-6m}(22)^{3m}, \text{if } n \geq 6m. \end{cases}$$

Assume $\mathrm{bdir}(L)$ is a CFL. Then $L' := \mathrm{bdir}(L) \ \cap \ (03)^+(13)^+(22)^+$ is a CFL, and from above we have $L' = \{(03)^{2m}(13)^{4m}(22)^{3m} \ : \ m \geq 1\}$, which is not a CFL. □

As Georg Zetzsche has kindly pointed out to us, the operation bdir was studied previously by Jantzen and Petersen [3]; they called it "twist". They proved our Theorems 20 and 21.

We now return to the proof of Theorem 13, which was postponed until now. We need two lemmas:

Lemma 22. *Suppose L is a regular language. Then $L' = \{ww^R \ : \ w \in L\}$ is a CFL.*

Proof. On input x, a PDA can guess w and verify it is in L, while pushing it on the stack. Nondeterministically it then guesses it is at the end of w and pops the stack, comparing to the input. □

Lemma 23. *For all words w we have* $w \sqcup\!\sqcup w^R = \mathrm{bdir}(w)\,\mathrm{bdir}(w)^R$.

Proof. If w is of even length then

$$\begin{aligned} w \sqcup\!\sqcup w^R &= (\mathrm{fh}(w)\mathrm{lh}(w)) \sqcup\!\sqcup (\mathrm{fh}(w)\mathrm{lh}(w))^R \\ &= (\mathrm{fh}(w)\mathrm{lh}(w)) \sqcup\!\sqcup (\mathrm{lh}(w)^R\mathrm{fh}(w)^R) \\ &= (\mathrm{fh}(w) \sqcup\!\sqcup \mathrm{lh}(w)^R)(\mathrm{lh}(w) \sqcup\!\sqcup \mathrm{fh}(w)^R) \\ &= \mathrm{bdir}(w)\mathrm{bdir}(w)^R. \end{aligned}$$

A similar proof works for w of odd length. □

We can now prove Theorem 13.

Proof. From Lemma 23 we have

$$\mathrm{pssr}(L) = \bigcup_{x \in L} x \sqcup\!\sqcup x^R = \bigcup_{x \in L} \mathrm{bdir}(x)\,\mathrm{bdir}(x)^R = \bigcup_{x \in \mathrm{bdir}(L)} xx^R.$$

If L is regular, then $\mathrm{bdir}(L)$ is regular, by Theorem 20. Then, from Lemma 22, it follows that $\mathrm{pssr}(L)$ is a CFL. □

7 Acknowledgment

We are grateful to Georg Zetzsche for his remarks.

References

[1] J. Berstel. *Transductions and Context-Free Languages*. Teubner, 1979.

[2] J. Erickson. How hard is unshuffling a string? `http://cstheory.stackexchange.com/questions/34/how-hard-is-unshuffling-a-string`, August 16 2010.

[3] M. Jantzen and H. Petersen. Cancellation in context-free languages: enrichment by reduction. *Theoret. Comput. Sci.* **127** (1994), 149–170.

[4] A. Mansfield. An algorithm for a merge recognition problem. *Disc. Appl. Math.* **4** (1982), 193–197.

[5] A. Mansfield. On the computational complexity of a merge recognition problem. *Disc. Appl. Math.* **5** (1983), 119–122.

[6] M. K. Warmuth and D. Haussler. On the complexity of iterated shuffle. *J. Comput. Sys. Sci.* **28** (1984), 345–358.