Information and Computation 246 (2016) 30–42

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

On bounded languages and reversal-bounded automata

Oscar H. Ibarra [a,*,1], Bala Ravikumar [b]

[a] Department of Computer Science, University of California Santa Barbara, CA 93106, USA
[b] Department of Computer & Engineering Science, Sonoma State University, Rohnert Park, CA 94928, USA

ARTICLE INFO

Article history:
Received 28 May 2013
Available online 2 December 2015

Keywords:
Context-free language (CFL)
Nondeterministic pushdown automaton (PDA)
Reversal-bounded
Semilinear set
Stratified linear set

ABSTRACT

Bounded context-free languages have been investigated for nearly fifty years, yet they continue to generate interest as seen from recent studies. Here, we present a number of results about bounded context-free languages. First we give a new (simpler) proof that every context-free language $L \subseteq w_1^* w_2^* \dots w_n^*$ can be accepted by a PDA with at most $2n-3$ reversals. We also introduce new collections of bounded context-free languages and present some of their interesting properties. Some of the properties are counter-intuitive and may point to some deeper facts about bounded CFLs. We present some results about semilinear sets and also present a generalization of the well-known result that over a one-letter alphabet, the families of context-free and regular languages coincide.

1. Introduction

The class of context-free languages (CFL) is one of the most important families of languages because of the nice mathematical properties they exhibit and because of their wide-ranging applications. Bounded CFLs [3] are interesting since they admit faster parsing algorithms and many problems that are undecidable for general CFLs are decidable for the bounded CFLs. Also many well-known examples and counter-examples for CFLs are bounded, for example, the standard example of an inherently ambiguous language is a bounded CFL [2]. Some recent works on bounded CFLs include [9,7] etc. Here we present some properties of bounded context-free languages.

We first show that every context-free language $L \subseteq w_1^* w_2^* \dots w_n^*$ can be accepted by a PDA with at most $2n-3$ reversals. This result was also recently shown by [9], but our proof is simpler and is based on a PDA while their proof is based on context-free grammars. Then, we introduce a number of bounded languages and present many of their properties. Some of these observations are unexpected. For example, listed below are three languages:

$$B_1 = \{a_1^{r_1} a_2^{r_2} a_3^{r_3} \mid r_1 + r_2 \geq r_3,\ r_2 + r_3 \geq r_1,\ r_3 + r_1 \geq r_2\}$$
$$B_2 = \{a_1^{n_1+n_2} a_2^{n_2+n_3} a_3^{n_3+n_1} \mid n_1, n_2, n_3 \geq 0\}$$
$$B_3 = \{a_1^{r_1} a_2^{r_2} a_3^{r_3} a_4^{r_4} \mid r_1 + r_3 = r_2 + r_4\}$$

Which of the above languages and/or their complements are context-free? What is the (minimum) number of reversals that are needed to accept them? The reader can check their intuition by looking at Section 4 where several such languages are introduced and their properties are discussed.

* Corresponding author.
E-mail addresses: ibarra@cs.ucsb.edu (O.H. Ibarra), ravi@cs.sonoma.edu (B. Ravikumar).
1 Supported in part by NSF Grants CCF-1143892 and CCF-1117708.

http://dx.doi.org/10.1016/j.ic.2015.11.007
0890-5401/

In Section 5, we briefly study the class of semilinear languages and provide a characterization for it in terms of reversal-bounded counter machines. In Section 6, we present a generalization of the well-known result that over a one-letter alphabet, the families of context-free and regular languages coincide. We also present some results about multitape PDAs with reversal bounded counters. We conclude with some open problems in Section 7.

2. Preliminaries

Let N be the set of natural numbers and $n \geq 1$. $Q \subseteq N^n$ is a *linear set* if there is a vector c in N^n (the constant vector) and a set of periodic vectors $V = \{v_1, \ldots, v_r\}$, $r \geq 0$, each v_i in N^n such that $Q = \{c + t_1 v_1 + \cdots + t_r v_r \mid t_1, \ldots, t_r \in N\}$. We denote this set as $Q(c, V)$. A finite union of linear sets is called a *semilinear set*.

A linear set $Q(c, V) \subseteq N^n$ is said to be *stratified* if:

1. Every $v \in V$ has at most two nonzero components, and
2. There exist no integers i, j, k, l with $1 \leq i < j < k < l \leq n$ and no vectors $u = (u_1, \ldots, u_n)$ and $v = (v_1, \ldots, v_n)$ in V such that $u_i v_j u_k v_l \neq 0$.

A finite union of stratified linear sets is called a *stratified semilinear set*.

Let $\Sigma = \{a_1, \ldots, a_n\}$. For $w \in \Sigma^*$, let $|w|$ be the number of letters (symbols) in w, and $|w|_{a_i}$ denote the number of occurrences of a_i in w. The *Parikh map* $P(w)$ of w is the vector $(|w|_{a_1}, \ldots, |w|_{a_n})$; similarly, the Parikh image of a language L is defined as $P(L) = \{P(w) \mid w \in L\}$.

It is known that the Parikh map of a language L accepted by a PDA (i.e., L is context-free) is an effectively computable semilinear set [10]. A useful generalization of this result for showing the decidability of a wide-range of problems is to extend this result to a larger class of machines. One such class is the class of PDAs augmented with reversal-bounded counters. A machine M in this class has a pushdown stack, together with a finite set of counters. Each counter can store a single symbol a other than the bottom of the stack symbol. At each move, based on the symbol read by the input head, and the symbol on the top of stack, and counter status (being 0 or nonzero) of each counter, the machine can choose one of a finite number of choices and execute it. This involves applying a move on the input tape (either stay or move right one position), change the state, remove the top of the stack symbol and push a string on the stack, and update each counter (by adding 0, +1 or −1 a's). The fact that the counter is reversal-bounded means that the number of times the counter can change from increasing mode (during which the counter value never decreases) to decreasing mode (during which the counter value never increases) is bounded by a constant, independent of the input length. Note that we place no such restriction on the pushdown stack. Acceptance of the input is by reaching a configuration in which the state is an accepting state the counters are 0.

The following result was shown in [5]:

Theorem 1.

1. *If $L \subseteq \Sigma^*$ is accepted by a PDA with reversal-bounded counters, then $P(L)$ is an effectively computable semilinear set.*
2. *If $L \subseteq w_1^* \cdots w_n^*$ is accepted by a PDA with reversal-bounded counters (where $w_1, \ldots, w_n$ are nonnull strings), then $Q_L = \{(i_1, \ldots, i_n) \mid w_1^{i_1} \cdots w_n^{i_n} \in L\}$ is an effectively computable semilinear set.*

A language L is *letter-bounded* if it is a subset of $a_1^* \cdots a_n^*$ for some distinct letters (symbols) $a_1, \ldots, a_n$. L is *bounded* if it is a subset of $w_1^* \cdots w_n^*$ for some (not necessarily distinct) nonnull strings $w_1, \ldots, w_n$. The following characterizations of letter-bounded context-free languages (CFLs) are from [3].

Theorem 2.

1. *Let $\Sigma = \{a_1, \ldots, a_n\}$, $n \geq 3$. Each CFL $L \subseteq a_1^* \cdots a_n^*$ is a finite union of sets of the following form:*

$$M(D, E, F) = \{a_1^i x y a_n^j \mid a_1^i a_n^j \in D, x \in E, y \in F\},$$

where $D \subseteq a_1^* a_n^*$, $E \subseteq a_1^* \cdots a_q^*$, $F \subseteq a_q^* \cdots a_n^*$, $1 < q < n$, are CFLs. Conversely, each finite union of sets of the form $M(D, E, F)$ is a CFL $L \subseteq a_1^* \cdots a_n^*$.
2. *A language $L \subseteq a_1^* \cdots a_n^*$, $n \geq 2$, is a CFL if and only if its Parikh map $P(L) \subseteq N^n$ is a stratified semilinear set (i.e., a finite union of stratified linear sets).*

Many restrictions of PDA have been considered in the literature [3,2] etc. One of the fundamental variation is a reversal-bounded PDA each of which is has an associated constant integer $k \geq 0$ such that on any accepting computation, the number of reversals of the stack is bounded by k. The class of languages accepted by reversal-bounded PDAs is known as *ultra-linear* languages. Another restriction is that of an unambiguous PDA that has exactly zero or one accepting computation on any string. The latter strings form the language accepted by such a PDA. The languages accepted by this class of PDA is the well-known class *unambiguous* context-free languages.

3. Characterization of bounded CFLs by reversal-bounded PDAs

We give two constructions to show that every CFL $L \subseteq a_1^* \cdots a_n^*$ can be accepted by a reversal-bounded PDA. The first construction is simple, but yields an upper bound of $2^n - 1$ on stack reversals. The second construction gives an upper bound of $2n - 3$ on the reversals and there are examples for which the construction achieves this bound.

We begin with the following lemma which is easily verified (see Corollary 2 for a stronger result).

Lemma 1. *Every context-free language $L \subseteq a_1^* a_2^*$ can be accepted by a 1 reversal PDA.*

Theorem 3. *Every CFL $L \subseteq a_1^* \ldots a_n^*$ can be accepted by a $2^{n-1} - 1$ reversal-bounded PDA.*

Proof. We will prove the claim by induction on n. Obviously, the theorem holds for $n = 1$ and also for $n = 2$ since, by Lemma 1, L can be accepted by 1 reversal-bounded PDA.

Now suppose $n \geq 3$. By Theorem 2, part 1, every CFL $L \subseteq a_1^* \cdots a_n^*$ is a finite union of sets of the form: $M(D, E.F) = \{a_1^i x y a_n^j \mid a_1^i a_n^j \in D, x \in E, y \in F\}$, where D, E, F are as defined in Theorem 2, part 1.

Now by Lemma 1, D can be accepted by a 1-reversal PDA M_1. By induction hypothesis, E and F can be accepted by PDAs M_2 and M_3 with at most $2^{n-2} - 1$ reversals because each of them is $n - 1$ letter-bounded.

We can build a PDA M for each $M(D, E, F)$. We assume that all the PDAs accept by empty-stack. M starts simulating M_1, and while still reading a_1's, it remembers the current state of M_1 in finite control, then places a \$ symbol on the stack and starts simulating M_2. After M_2 accepts by empty-stack, the \$ symbol is reached on the stack. At this point, M starts simulating M_3. When the \$ symbol is reached again, M continues the simulation of M_1 from the state it remembered until the string is accepted. Clearly, the total number of reversals is at most $2^{n-2} - 1 + 2^{n-2} - 1 + 1 = 2^{n-1} - 1$. $\square$

We now give a construction that improves the upper bound on the stack reversals.

Theorem 4. *Let $L \subseteq a_1^* \cdots a_n^*$ be a CFL given by a PDA. Then we can construct a $2n - 3$ reversal-bounded PDA accepting L. Moreover, there are examples for which the construction achieves the bound $2n - 3$ for every $n \geq 2$.*

Proof. From Theorem 2, part 2, $L \subseteq a_1^* \cdots a_n^*$ is a CFL if and only if its Parikh map $P(L)$ is a stratified semilinear set (i.e., finite union of stratified linear sets.) Since r reversal-bounded PDA languages (for any $r \geq 0$) are closed under union, it is sufficient to show that a language L whose Parikh map is a stratified linear set is accepted by a $2n - 3$ reversal-bounded PDA.

Let Q be a linear set generated by the constant vector $c = (c_1, \ldots, c_n)$ and a set of basic vectors in the set V.
For $1 \leq i \leq n$, let $V_i = \{v \mid v \in V,$ and v has only one nonzero component: the ith component$\}$.
For $1 \leq i, j \leq n$, $i \neq j$, let $V_{ij} = \{v \mid v \in V,$ and v has two nonzero components: the ith and jth components$\}$.
We construct a PDA M that accepts $L = \{a_1^{i_1} \cdots a_n^{i_n} \mid (i_1, \ldots, i_n) \in Q\}$. In addition to the bottom of the stack symbol, M has a pushdown symbol T_{ij} for $1 \leq i < j \leq n$.
Given $a_1^{i_1} \cdots a_n^{i_n}$, M processes each a_i-segment, for $i = 1, \ldots, n$, as follows:

1. M reads c_i a_i's on the input, where c_i is the ith component of the constant vector $c = (c_1, \ldots, c_n)$ of Q.
2. If $V_i = \varnothing$, this step is skipped.
   For each $v \in V_i$ if $v = (0, \ldots, 0, d_i, 0, \ldots, 0)$ (where d_i is the ith component), then for nondeterministically chosen $t_v \geq 0$, M does the following t_v times: reads d_i a_i's without changing the stack. (Thus, for each $v \in V_i$, M reads a total of $t_v d_i$ a_i's.)
3. If there are no vectors in V with nonzero components in position i and in position less than i, this step is skipped. In particular, when $i = 1$, this step is skipped.
   Suppose $V_{j_1 i}, \ldots, V_{j_t i}$ are nonempty and $1 \leq j_1 < \ldots < j_t < i$. Then M does the following for $k = t, t-1, \ldots, 1$ in this order:
   For each $v \in V_{j_k i}$, if $v = (0, \ldots, 0, d_{j_k}, 0, \ldots, 0, d_i, 0, \ldots, 0)$ (i.e., the nonzero components are in the j_kth and ith positions), then M pops the stack such that for each $T_{j_k i}$ it pops, it reads an a_i. (Thus, for each $v \in V_{j_k i}$, M pops a total of $t_v d_{j_k}$ $T_{j_k i}$'s that have been stored earlier and reads a total of $t_v d_{j_k}$ a_i's.)
4. If there are no vectors in V with nonzero components in position i and in position greater than i, this step skipped.
   Suppose $V_{i j_1}, \ldots, V_{i j_t}$ are nonempty and $i < j_1 < \ldots < j_t$. Then M does the following for $k = t, t-1, \ldots, 1$ in this order:
   For each $v \in V_{i j_k}$, if $v = (0 \ldots, 0, d_i, 0, \ldots, 0, d_{j_k}, 0, \ldots, 0)$ (i.e., the nonzero components are in positions i_{th} and j_kth), then for nondeterministically chosen $t_v \geq 0$, M does the following: reads d_i a_i's and stack d_{j_k} $T_{i j_k}$'s on the pushdown. (Thus, for each $v \in V_{i j_k}$, M reads a total of $t_v d_i$ a_i's and stacks a total of $t_v d_{j_k}$ $T_{i j_k}$'s on the pushdown.)

After the four steps above, M has completed processing the a_i-segment and should be reading the first a_{i+1}. (If not, M rejects the input and halts.) M then proceeds to process the a_{i+1}-segment. When all the a_i-segments have been processed successfully, M accepts the input and halts.

We will now sketch a proof that M accepts L. The basic idea is the following: let $w = a_1^{i_1} a_2^{i_2} \ldots a_n^{i_n}$ be a string in L. Then $(i_1, i_2, \ldots, i_n) = c + p_1 v_1 + \ldots + p_m v_m$, where c is the constant vector and $v_1, \ldots, v_m$ are the vectors that form the basis. This involves checking that $i_j = c_j + \sum_{r=1}^{m} p_r v_{jr}$, where c_r is the r-th component of c and v_{jr} is the r-th component of v_j. Since V is stratified, there are at most two nonzero components in any vector v_j.

M verifies the above equation by guessing the numbers $p_1, \ldots, p_m$ and keeping the expression $\sum_{r=1}^{m} p_r v_{jr}$ on the stack while reading the first nonzero component of the vectors whose second nonzero component is j, and matching this value with the value i_j by popping one symbol for each input symbol a_j read. The condition (2) of stratified linear set is guaranteed by the fact that there will be no intervening symbols on the stack when popping. The vectors with only one nonzero component are handled using the finite-control.

Next, let us determine the number of stack reversals M makes on an input string in $a_1^* \cdots a_n^*$. For each a_i-segment, $2 \le i \le n-1$, M may make a sequence of pops followed by a sequence of pushes. Hence, the number of alternations from popping to pushing and vice-versa for these $n-2$ segments is $2n-5$. (Note that if $n = 2$, the number is 0.) Now the a_1-segment and the a_n segments contribute a pushing sequence and a popping sequence. Thus, in total, there can be $2n - 5 + 2 = 2n - 3$ alternations between popping and pushing and vice-versa or, equivalently, M makes $2n-3$ stack reversals. □

To see that the $2n-3$ bound on the stack reversal is achievable, consider the CFL $L_{cycle}^n = \{a_1^{i_1+i_2} a_2^{i_2+i_3} \cdots a_{n-1}^{i_{n-1}+i_n} a_n^{i_1+i_n} \mid i_1, \ldots, i_n \ge 0\}$. Clearly, the Parikh map of L_{cycle}^n is a stratified linear set with constant vector $c = (0, \ldots, 0)$ and set of periodic vectors $V = \{(1, 0, \ldots, 0, 1), (1, 1, 0, \ldots, 0), (0, 1, 1, 0, \ldots, 0), (0, 0, 1, 1, 0, \ldots, 0), \ldots, (0, \ldots, 0, 1, 1, 0), (0, \ldots, 0, 1, 1)\}$. It is easy to verify that the PDA accepting L_{cycle}^n using the construction described above will make $2n-3$ reversals. In fact, later we will show that this bound cannot be improved in general since there are n-bounded CFLs that cannot be accepted by a PDA with fewer than $(2n-3)$ reversals.

Corollary 1. *Let $L \subseteq w_1^* \cdots w_n^*$ be a CFL, where $w_1, \ldots, w_n$ are nonnull strings. Then L can be accepted by a $2n-3$ reversal-bounded PDA, and this reversal-bound is tight.*

Proof. Suppose L is accepted by a PDA M. Let $a_1, \ldots, a_n$ be distinct symbols. We can easily construct a PDA M_1 accepting $L_1 = \{a_1^{i_1} \cdots a_n^{i_n} \mid w_1^{i_1} \cdots w_n^{i_n} \in L\}$. ($M_1$ on a_i simulates the steps of M on string w_i.) Then from Theorem 4, we can construct a $2n-3$ reversal-bounded PDA M_2 accepting L_1. Finally, we can construct from M_2 a $2n-3$ reversal-bounded PDA accepting L. □

For the case when $n = 2$, we have:

Corollary 2. *Every CFL $L \subseteq w_1^* w_2^*$ can be accepted by a 1-reversal counter machine. (Here a counter machine refers to a PDA in which the stack alphabet consists of two symbols one of which is used only as the bottom of the stack symbol.)*

Proof. When $n = 2$, the PDA constructed in the proof of Theorem 4 has only one stack symbol T_{12}, in addition to the bottom of the stack symbol. Hence the pushdown stack is actually a counter. □

Malcher and Pighizzini [9] show that the $2n-3$ reversal-bound in Theorem 4 is tight for a different candidate family L_k:

Theorem 5. *For any $k \ge 1$, $L_k = \{a_1^{i_1+i_2} a_2^{i_2+i_3} \cdots a_{n-1}^{i_{n-1}+i_n} a_n^{i_n} \mid i_1 = 2^k, i_2, \ldots, i_n \ge 1\}$ cannot be accepted by any PDA in less than $2n-3$ reversals.*

We note that the result in [9] gave a lower bound of $n-1$ turns which, when using our definition of reversal-bound, corresponds to the lower bound of $2n-3$. (Basically, they count only a down-turn as a reversal, while we count both up-turns and down-turns as reversals.)

4. L_{cycle}^n and related languages

The language $L_{cycle}^n = \{a_1^{i_1+i_2} a_2^{i_2+i_3} \ldots a_{n-1}^{i_{n-1}+i_n} a_n^{i_n+i_1} \mid i_1, \ldots, i_n \ge 0\}$ has some interesting characteristics, which we explore in this section. A related language has been studied in [9].

We first introduce some notation. For an odd integer $n \ge 3$, let $C = (p_1, \ldots, p_n)$, and

$$C_1 = (p_1, p_2, \ldots, p_n)$$

$$C_2 = (p_2, p_3, \ldots, p_n, p_1)$$

$$C_3 = (p_3, p_4, \ldots, p_n, p_1, p_2)$$

$$\vdots$$

$$C_i = (p_i, p_{i+1}, \ldots, p_n, p_1, \ldots, p_{i-1})$$

Thus, C_i is the i-th circular shift of C.

For a list $C = (p_1, p_2, \ldots, p_n)$, denote by $S(C) = p_1 - p_2 + p_3 - \cdots + (-1)^{n-1} p_n$. For example, $S((3, 6, 7)) = 3 - 6 + 7 = 4$. Consider the following four collections of languages:

1. $L^n_{cycle} = \{a_1^{i_1+i_2} a_2^{i_2+i_3} \ldots a_{n-1}^{i_{n-1}+i_n} a_n^{i_n+i_1} \mid i_1, \ldots, i_n \geq 0\}$, for any $n \geq 2$.
2. $L^n_1 = \{a_1^{p_1} a_2^{p_2} \ldots a_n^{p_n} \mid p_1 + \ldots + p_n$ is even, and $S(C) = 0$ where $C = (p_1, \ldots, p_n)\}$, for any even $n \geq 2$.
3. $L^n_2 = \{a_1^{p_1} a_2^{p_2} \ldots a_n^{p_n} \mid p_1 + \ldots + p_n$ is even, $S(C_i) \geq 0$ for $1 \leq i \leq n\}$, for any odd $n \geq 3$ where $C = (p_1, \ldots, p_n)$.
4. $L^n_3 = \{a_1^{p_1} a_2^{p_2} \ldots a_n^{p_n} \mid S(C_i) \geq 0$ for $1 \leq i \leq n\}$, for any odd $n \geq 3$ where $C = (p_1, \ldots, p_n)$. Note that the condition $p_1 + \ldots + p_n$ is even is no longer assumed in L^n_3.

We will show the following in this section:

1. For $n = 2$ and 4: $L^n_{cycle} = L^n_1$, and it and its complement can be accepted by a $2n - 3$ reversal-bounded deterministic counter machine.
2. For any even $n \geq 6$, $L^n_{cycle} \neq L^n_1$. (Thus the equivalence $L^n_{cycle} = L^n_1$ holds only for $n = 2$ and 4.)
3. For any odd $n \geq 3$, $L^n_{cycle} = L^n_2$, and it can be accepted by an unambiguous PDA (but not likely by any counter machine, even if it is allowed to be ambiguous and there is no restriction on the reversals).
4. For any odd $n \geq 3$, the complement of L^n_{cycle} (= complement of L^n_2) can be accepted by a $2n - 3$ reversal-bounded counter machine.
5. For any odd $n \geq 3$, L^n_3 can be accepted by an unambiguous $2n - 3$ reversal bounded PDA.
6. For any odd $n \geq 3$, the complement of L^n_3 can be accepted by a $2n - 3$ reversal-bounded counter machine.

First we consider even n and determine the connection between L^n_{cycle} and L^n_1.

Lemma 2. *For $n = 2$ and 4, $L^n_{cycle} = L^n_1$.*

Proof. For $n = 2$, the claim is obvious since $L^n_{cycle} = L^2_1 = \{a^n b^n | n \geq 0\}$. Let $w = a^p b^q c^r d^s \in L^4_{cycle}$. (Note that we use a, b, c, d instead of a_1, etc. so that it is easier to read.) Then there exist $i, j, k, m \geq 0$ such that $p = i + j$, $q = j + k$, $r = k + m$ and $s = m + i$. It is easy to see that $p + r = i + j + k + m = q + s$, and so $w \in L^4_1$.

Conversely, let $w \in L^4_1$. Then, $w = a^p b^q c^r d^s$ for some p, q, r and s (all nonnegative) such that $p + r = q + s$. We will show that there exist $i, j, k, m \geq 0$ such that $p = i + j$, $q = j + k$, $r = k + m$ and $s = m + i$. There are two cases to consider.

Case 1. $r > s$. In this case, let $i = 0$, $j = p$, $m = s$, $k = r - s$. Thus, $w \in L^{(4)}_{cycle}$.

Case 2. $r \leq s$. We first note that $s - r \leq p$ and $s - r \leq s$. The former inequality follows from: $s - r = p - q \leq p$ since $q \geq 0$. The second inequality follows from $r \geq 0$. Combining the two inequalities, we have $s - r \leq \min\{p, s\}$.

Now we describe the choice of i, j, k and m: choose i such that $s - r \leq i \leq \min\{p, s\}$, $j = p - i$, $k = r - s + i$ and $m = s - i$. From the inequality above, it should be clear that $i, j, k, m \geq 0$. Hence, $w \in L^4_{cycle}$. □

For $n = 2$ and 4, L^n_{cycle} $(= L^n_1)$ is context-free. In fact, it and its complement can be accepted by $2n - 3$ reversal-bounded deterministic 1-counter PDA, as shown in the next theorem.

Theorem 6. *For any even $n \geq 2$, L^n_1 and $\overline{L^n_1}$ can be accepted by $2n - 3$ reversal-bounded deterministic 1-counter PDA.*

Proof. We describe a deterministic counter machine accepting L^n_1. To check the condition $p_1 - p_2 + p_3 - p_4 + p_5 \ldots - p_n = 0$, the machine proceeds as follows. After it has processed the first j blocks, the PDA stores in the counter the value of $x = p_1 - p_2 + \ldots + (-1)^{j-1} p_j$ if $x > 0$, or it holds the value of $-x$. It also remembers in the finite control the sign status – namely whether it is holding the prefix sum x or $-x$. Now suppose the next block is an odd block. If the current value in stack is x, then it starts pushing while processing the next block all the way. If it holding $-x$, then it starts popping until the stack becomes 0, then starts pushing with the sign status reversed. It does the exact opposite for the even numbered block. At the end, it accepts if and only if the counter becomes 0. It is clear that this machine is deterministic and makes at most $2n - 3$ reversals. It also follows that $\overline{L^n_1}$ can be accepted by a $2n - 3$ reversal-bounded deterministic counter machine. □

For $n = 6$ (and larger n), the situation is different as the following shows.

Proposition 1. *For any even $n \geq 6$, $L^n_{cycle} \neq L^n_1$.*

Proof. Consider the string $w = a_1^{18}a_2{}^7a_3{}^3a_4{}^{11}a_5{}^2a_6{}^5$. It is easy to see that $w \in L^6_1$. We will see that $w \notin L^6_{cycle}$. The reason is simple: For w to be in L^6_{cycle}, there should exist nonnegative integers $i_1, \ldots, i_6$ such that $i_1 + i_2 = 18$, $i_2 + i_3 = 7, \ldots, i_6 + i_1 = 5$. Now since $i_2 + i_3 = 7$, $i_2 \leq 7$ and so $i_1 \geq 11$. However, $i_6 + i_1 = 5$, so $i_1 \leq 5$. This is a contradiction. Similar counterexamples can be constructed for larger even values of n. $\square$

Now, we consider L^n_2 for odd $n \geq 3$.

Lemma 3. *For any odd $n \geq 3$, $L^n_{cycle} = L^n_2$.*

Proof. We will first show the claim for $n = 3$. Note that $L^3_{cycle} = \{a^{i+j}b^{j+k}c^{k+i} \mid i, j, k \geq 0\}$ and $L^3_2 = \{a^pb^qc^r \mid p + q + r$ is even, $p+q \geq r$, $q+r \geq p$, $r+p \geq q\}$. Let $w = a^pb^qc^r \in L^3_{cycle}$. Then, there exist integers $i, j, k \geq 0$ such that $p = i+j$, $q = j+k$ and $r = k+i$. Solving for i, j, k in terms of p, q and r, we get: $i = \frac{(p+r-q)}{2}$, $j = \frac{(p+q-r)}{2}$, and $k = \frac{(r+q-p)}{2}$. Since $i, j, k \geq 0$, it follows that $p + q \geq r$, $q + r \geq p$, $r + p \geq q$. Since $p + q + r$ is $2(i + j + k)$ it is also obvious that $p + q + r$ is even. Hence, $w \in L^3_2$.

Conversely, let $w = a^pb^qc^r \in L^3_2$. Then, we can choose i, j, k to be $i = \frac{(p+r-q)}{2}$, $j = \frac{(p+q-r)}{2}$, and $k = \frac{(r+q-p)}{2}$. From the condition that $p+q \geq r$, it follows that $j \geq 0$. From the condition that $p+q+r$ is even, it follows that $\frac{(p+q-r)}{2}$ is an integer and so j is a nonnegative integer. Similarly, it follows that i and k are nonnegative integers. Hence, $w \in L^3_{cycle}$.

Generalization for arbitrary n is easy: Let $w = a_1^{p_1}a_2^{p_2}\cdots a_n^{p_n} \in L^n_{cycle}$. Then there exist $i_1, i_2, \ldots, i_n$ such that $p_1 = i_1 + i_2$, $p_2 = i_2 + i_3, \ldots, p_{n-1} = i_{n-1} + i_n$, $p_n = i_n + i_1$. As in the 3-bounded case, solving for the i_j's in terms of the p_j's and noting that the i_j's are nonnegative, we can check that the p_j's satisfy the conditions in the definition of L^n_2.

For the converse, let $w = a_1^{p_1}a_2^{p_2}\ldots a_n^{p_n} \in L^n_2$. Define $C = (p_1, p_2, \ldots, p_n)$. Then we choose $i_1 = S(C_1)/2, i_2 = S(C_2)/2, \ldots, i_n = S(C_n)/2$. Then, as in the 3-bounded case, it can be verified that $w \in L^n_{cycle}$. $\square$

From the above lemma and Theorem 4, we have:

Theorem 7. *For any odd $n \geq 3$, $L^n_2(= L^n_{cycle})$ can be accepted by a $2n - 3$ reversal-bounded unambiguous PDA.*

Next, we show that the complement of L^n_{cycle} can be accepted by a reversal-bounded counter machine.

Theorem 8. *For odd $n \geq 3$, $\overline{L^n_2}$ $(= \overline{L^n_{cycle}})$ can be accepted by a $2n - 3$ reversal-bounded counter machine, M.*

Proof. We construct M to accept the complement of L^n_2. Again, let $C = (p_1, p_2, \ldots, p_n)$. Let w be an input to M. Clearly, M's finite-control can check and accept if w is not in $a_1^* \cdots a_n^*$ or if $w = a_1^{p_1}\cdots a_n^{p_n}$ but $p_1 + \cdots + p_n$ is not even. Now we describe the operation of M when the input has the correct format, but is not in L^n_2. Clearly, w is not in L^n_2 if there exists an $1 \leq i \leq n$ such that $S(C_i) < 0$. So M simply guesses an i, which gives an expression of the p_j's with arithmetic operations minus and plus, but can be rewritten so that $p_1, p_2, \ldots, p_n$ appear in this order. (For example, for $n = 5$, $S(C_3) = p_3 - p_4 + p_5 - p_1 + p_2 = -p_1 + p_2 + p_3 - p_4 + p_5$.) Thus, we can associate with each $S(C_i)$ a sign vector. For example for $n = 5$ the signs vectors associated with:

$$S(C_1) = p_1 - p_2 + p_3 - p_4 + p_5 \text{ is } (+, -, +, -, +)$$

$$S(C_2) = p_2 - p_3 + p_4 - p_5 + p_1 \text{ is } (+, +, -, +, -)$$

$$S(C_3) = p_3 - p_4 + p_5 - p_1 + p_2 \text{ is } (-, +, +, -, +)$$

etc.

The sign vectors are built into the finite-control of M. So when M guesses an i, it knows the associated vector, e.g., if it guesses $S(C_3)$, the sign vector $(-, +, +, -, +)$ indicates that the expression to evaluate is $-p_1 + p_2 + p_3 - p_4 + p_5$. Then M scans the input and evaluates the expression deterministically. In order to compute the expression, the machine has to remember the status as either P or N. P means Increment (Decrement) on seeing a term with positive (negative) sign. For N, it is the opposite. When the counter becomes empty in the middle of a block, switch from P to N (and N to P). The input string is accepted when the value held by the counter is negative when the entire input has been processed.

We note that $S(C_1)$ is the only guess that will require $2n - 3$ reversals. All the others will require fewer reversals. $\square$

Finally, we consider the language L_3^n:

Theorem 9. *For any odd $n \geq 3$:*

1. *L_3^n can be accepted by a $2n-3$ reversal-bounded unambiguous PDA.*
2. *$\overline{L_3^n}$ can be accepted by a $2n-3$ reversal-bounded counter machine. Further, for $n = 3$, the counter machine can be made unambiguous.*

Proof. To prove (1): on input $w = a_1^{p_1} a_2^{p_2} \ldots a_n^{p_n}$, M' simulates M of Theorem 7 on input $w = a_1^{2p_1} a_2^{2p_2} \ldots a_n^{2p_n}$. M' accepts if M accepts. It should be clear that M' accepts L_3^n and is $2n-3$ reversal-bounded.

To prove part (2), we construct a $2n-3$ reversal-bounded counter machine M'' to accept $\overline{L_3^n}$ as follows. Let w be an input to M''. The finite-control can check and accept if w is not in $a_1^* \cdots a_n^*$. If the input has the correct format, M'' checks that $S(C_i) < 0$ for some $1 \leq i \leq n$, as in the proof of Theorem 8.

It can be seen that the counter machine described above is unambiguous for $n = 3$ as follows: Note that $\overline{L_3^n} = L_1 \cup L_2 \cup L_3$ where

$$L_1 = \{a_1^{i_1} a_2^{i_2} a_3^{i_3} \mid i_1 + i_2 < i_3\},$$
$$L_2 = \{a_1^{i_1} a_2^{i_2} a_3^{i_3} \mid i_2 + i_3 < i_1\}, \text{ and}$$
$$L_3 = \{a_1^{i_1} a_2^{i_2} a_3^{i_3} \mid i_3 + i_1 < i_2\}$$

On a given input w, the counter machine guesses that $w \in L_j$ for $j = 1, 2$ or 3 and checks this fact. Since each of the languages L_j is deterministic, the only nondeterminism is the initial guess. However, note that the languages L_1, L_2 and L_3 are pairwise disjoint so no string w belongs to more than one of the above languages. Thus for each input string w, there is at most one accepting computation.

However, the counter machines described above are ambiguous for larger odd values. We will provide an example to illustrate this for $n = 5$. Consider the string $w = a_1^{i_1} a_2^{i_2} a_3^{i_3} a_4^{i_4} a_5^{i_5}$ where $i_1 = 7$, $i_2 = 10$, $i_3 = 7$, $i_4 = 22$ and $i_5 = 8$. It is easy to see that in this case, $S(C_1) < 0$ and $S(C_3) < 0$. Therefore, the above counter machine is ambiguous. Similar examples can be constructed for larger odd values. $\square$

A natural question regarding the languages described in this section (namely, L_{cycle}^n, L_1^n, L_2^n etc.) is if the upper-bound of $2n-3$ on number of reversals presented in our construction above is tight. Recall Theorem 4 in which we presented the candidate language for which Malcher and Pighizzini [9] presented a technique for showing a lower-bound on the number of reversals. They established a tight-bound (matching the upper and lower-bounds) for L_k language that closely resembles L_{cycle}^n. Recall that L_k is:

$$L_k = \{a_1^{i_1+i_2} a_2^{i_2+i_3} \cdots a_{n-1}^{i_{n-1}+i_n} a_n^{i_n} \mid i_1 = 2^k, i_2, \ldots, i_n \geq 1\}.$$

Note that this language is remarkably similar to L_{cycle}^n. The primary difference is that in the former language (L_k), i_1 takes a constant value 2^k while in the latter language (L_{cycle}^n), i_1 is arbitrary and that the count of the last block is i_n while in the latter, it is the sum $i_n + i_1$. Another (minor) difference is that in the former language, the variables $i_2, \ldots, i_n$ are required to be greater than 0 while in the latter case, the variables are allowed to take 0 value. In view of the similarity between the two languages, one may guess that the lower-bound of $2n-3$ carries over to the latter language along with the proof technique.

However, the situation seems to be more subtle in that the proof technique of Malcher and Pighizzini does not extend to L_{cycle}^n. In fact, the bound $2n-3$ does not even hold for L_{cycle}^n. We will show this at least in the case of even $n > 2$. The smallest such case is $n = 4$ for which the upper-bound presented in Theorem 4 is 5. However, this is not tight. In the following, we will show that L_{cycle}^4 can be accepted with three reversals.

Claim. Let $L_{cycle}^4 = \{a_1^{n_1+n_2} a_2^{n_2+n_3} a_3^{n_3+n_4} a_4^{n_4+n_1} \mid n_1, n_2, n_3, n_4 \geq 0\}$. Then L_{cycle}^4 can be accepted by a PDA with 3 reversals.

Proof of Claim. Recall the obvious PDA construction that would require 5 reversals. PUSH $n_1 + n_2$ symbols while scanning a_1. Then, while scanning a_2, nondeterministically POP n_2 symbols, then PUSH n_3 symbols. Then, while scanning a_3, POP n_3 symbols, then PUSH n_4 symbols. Finally when scanning a_4, POP $n_4 + n_1$ symbols and reach the end of the input. This involves exactly 5 reversals.

Construction of a PDA that uses only 3 reversals for this language is as follows:

Recall from Lemma 2 that:

$$L_{cycle}^4 = L_1^4 = \{a_1^{p_1} a_2^{p_2} a_3^{p_3} a_4^{p_4} \mid p_1 + p_2 + p_3 + p_4 \text{ is even and } p_1 + p_3 = p_2 + p_4\}.$$

To show that L_1^4 can be accepted by a PDA with 3 reversals, we will write L_1^4 as a union of three languages $A_1 \cup A_2 \cup A_3$ where:

$$A_1 = \{a_1^n a_2^m a_3^m a_4^n \mid n, m \geq 0\}.$$
$$A_2 = \{a_1^{m+k} a_2^n a_3^{n-m} a_4^k \mid n, m, k \geq 0 \text{ and } n \geq m\}$$
$$A_3 = \{a_1^k a_2^{n-m} a_3^n a_4^{m+k} \mid n, m, k \geq 0 \text{ and } n \geq m\}$$

Claim 1. $L_1^4 = A_1 \cup A_2 \cup A_3$.

Proof of Claim 1. Let $z \in L_1^4$. Then, $z = a_1^{p_1} a_2^{p_2} a_3^{p_3} a_4^{p_4}$ for some nonnegative integers p_1, p_2, p_3 and p_4 such that $p_1 + p_2 + p_3 + p_4$ is even and $p_1 + p_3 = p_2 + p_4$.

Case 1. $p_1 = p_4$. Then, clearly $p_2 = p_3$. In this case, it is clear that $z \in A_1$.

Case 2: $p_1 > p_4$. It follows that $p_3 > p_2$. It is easy to see z is in A_2.

Case 3: $p_1 < p_4$. It follows that $p_2 > p_3$. It is clear that z is in A_3.

Thus $L_1^4 \subseteq A_1 \cup A_2 \cup A_3$. It is easy to see that $A_1 \cup A_2 \cup A_3 \subseteq L_1^4$. This completes the proof of Claim 1.

Claim 2. $A_1 \cup A_2 \cup A_3$ can be accepted by a 3-reversal PDA.

Proof of Claim 2. We will show that each A_i requires a PDA with at most 3 reversals. A_1 is easily seen to be linear and hence requires at most one reversal. We can design a 3-reversal PDA for A_2: On input $z = a_1^{p_1} a_2^{p_2} a_3^{p_3} a_4^{p_4}$, the PDA M guesses integers m, n and k and verifies that $p_1 = m + k$, $p_2 = n$ etc. as follows: it **pushes** $m + k = p_1$ symbols 1 on stack while scanning a_1, pops m symbols while scanning a_2. Then pushes a different symbol 2 for each of the remaining a_2 scanned. Now the stack has k 1's followed by $n - m$ 2's (from bottom to top). Now M checks that the number of a_3's on the input tape is equal to $n - m$ by **popping** a 2 for each a_3. Then, while scanning a_4, it continues to pop a 1 for each a_4. It accepts if the stack becomes empty just when the entire input has been read. Using a similar argument, it is easy to see that A_3 can be accepted by a 3-reversal PDA. □

We next generalize the above claim and show the following:

Theorem 10. *For any even $n \geq 2$, $L_1^n = \{a_1^{p_1} a_2^{p_2} \ldots a_n^{p_n} \mid p_1 + \ldots + p_n$ is even, $p_1 - p_2 + p_3 - \ldots - p_{n-2} + p_{n-1} - p_n = 0\}$ can be accepted by a 3-reversal PDA.*

Proof. We showed above that L_1^4 can be accepted by a PDA with 3 reversals. We will use this as a basis for an induction based proof. Instead of formally showing the induction proof, we will show it for L_1^6 and the idea extends to larger values of n in a very similar way.

Recall that $L_1^6 = \{a_1^{p_1} a_2^{p_2} \ldots a_6^{p_6} \mid p_1 + p_2 + \ldots + p_6$ is even and $p_1 + p_3 + p_5 = p_2 + p_4 + p_6\}$

It can be shown that L_1^6 can be written as a union of three languages A_1, A_2 and A_3 where:

$$A_1 = \{a_1^m w a_6^m \mid m \geq 0, w \in L_1^4\}.$$

Note that L_1^4 in the above is defined over the symbols $\{a_2, a_3, a_4, a_5\}$.

$$A_2 = \{a_1^{s_6+k} a_2^{s_2} a_3^{s_3} a_4^{s_4} a_5^{s_5} a_6^{s_6} \mid k + s_2 + s_4 = s_3 + s_5\}.$$
$$A_3 = \{a_1^{s_1} a_2^{s_2} a_3^{s_3} a_4^{s_4} a_5^{s_5} a_6^{k+s_1} \mid s_2 + s_4 = s_3 + s_5 + k\}.$$

The claim follows from the two observations below:

(a) $L_1^6 = A_1 \cup A_2 \cup A_3$ and
(b) Each A_j can be accepted by a 3 reversal PDA.

The proof of (a) is almost identical to that of the proof of Claim 1. The proof of (b) is as follows: Let M be a PDA for L_1^4. (Here an appropriate change has to be made, instead of accepting strings of the form $a_1^* \ldots a_4^*$, we will assume M accepts strings of the form $a_2^* \ldots a_5^*$. We also assume that when M enters an accepting state, its stack is empty.) We design a 3-reversal PDA M_j for A_j. Then the claim follows. As usual, we will assume that the input is of the form $a_1^* \ldots a_6^*$. M_1 guesses m and pushes m new stack symbols g_1 (new in the sense that g_1 is different from the stack symbols used by M) while reading m of a_1's, then it starts simulating M until it reaches an accepting state. It also checks that it is reading g_1 on the stack when M reaches accepting state. Then, for the remaining a_6's on the input tape, it pops g_1 and accepts the input if and only if the stack is empty exactly when the input end is reached. It is clear that M_1 accepts A_1. Also note that the number of reversals used by M_1 is the same as M since it only adds an extra push phase at the start and an extra pop at the end.

The PDA M_2 for A_2 is as follows: We make the same assumptions about M as in the previous paragraph. M_2, just as M_1 starts pushing s_6 copies of the new symbol g_1, one for each a_1 read. Then, it simulates M just as M_1 does, except for one change. It treats both a_1 and a_2 as a_2 symbol. The rest of the details are as in the above paragraph.

The construction for A_3 is very similar and so we omit the details. This concludes the proof. □

Next we will show that three reversals are necessary for accepting L_1^n for all $n > 2$.

Since it can be assumed that a PDA always starts with a PUSH phase and ends with a POP phase, the number of reversals is always an odd number and hence, if we can show that L_1^n is not linear then it follows that L_1^n requires three reversals. We need a lemma from [4].

Lemma 4. *Let $L \subseteq a^+b^+c^+c^+$ be such that*

(1) *for all $n, r \geq 1$, $a^n b^n c^r d^r \in L$,*
(2) *if $a^n b^n c^r d^s \in L$, then $r \leq s$, and*
(3) *there exist integers $t_1, t_2 \geq 1$ such that if $a^n b^m c^r d^s \in L$ for some $m < n$, then $(n-m)t_1 \leq (r+s)t_2$.*

Then, L is not a linear context-free language.

Theorem 11. *For every $n > 2$, a PDA accepting L_1^n requires three reversals.*

Proof. As remarked above, it suffices to show that L_1^n is not a linear CFL. We will first show this for $n = 4$. We define M_1^4, a language closely related to L_1^4 as follows: $M_1^4 = \{a_1^n a_2^m a_3^r a_4^s \mid n, m, r, s \geq 1,\ n + r = m + s\}$. It is easy to see that L_1^4 is a linear CFL if and only if M_1^4 is. It is easy to check that M_1^4 satisfies the conditions of Lemma 4. (The first two conditions are obvious. We can choose $t_1 = t_2 = 1$ so that condition (3) is satisfied.) This shows the claim for $n = 4$. To show that L_1^n is not linear for larger values n, it suffices to see that L_1^4 can be written as an intersection of L_1^n and a regular set. □

Finally, we note that an analog of Ogden's lemma for linear languages [1] can be used to show that L_{cycle}^3 is not linear.

5. Semilinear languages

A language $L \subseteq w_1^* \cdots w_n^*$ is called a *semilinear language* if the set $Q = \{(i_1, \ldots, i_n) | w_1^{i_1} \cdots w_n^{i_n} \in L\}$ is a semilinear set.

Corollary 3. *The class of semilinear languages over $w_1^* \ldots w_n^*$ is the closure under intersection of languages accepted by $2n-3$ reversal-bounded PDAs.*

Proof. In [8], it was shown that the class of semilinear sets is the least class of sets which contains all of the stratified semilinear sets and is closed under finite intersection. The result then follows from Corollary 1. □

Thus a language $L \subseteq w_1^* \ldots w_n^*$ is semilinear if and only if $L = \cap_{j=1}^m L_j$ for some languages $L_1, \ldots, L_m$ such that each L_j can be accepted by $2n-3$ reversal-bounded PDA.

From a generalization of Theorem 1, part 2 and the above corollary, we get:

Theorem 12. *The class of languages over $w_1^* \cdots w_n^*$ accepted by PDAs with reversal-bounded counters is the closure under intersection of languages accepted by $2n-3$ reversal-bounded PDAs.*

A resetting PDA is a special case of a two-way PDA (with input end markers). The machine starts with the input head on the left end marker with stack containing only a distinguished bottom of the stack symbol, which is never modified. The machine then computes like a PDA but when the input head reaches the right end marker, it either enters an accepting state eventually, or resets the input head to the left end marker in some state (which need not be the initial state) with stack again containing only the bottom of the stack symbol. The machine can then make another (one-way)sweep on the input like a PDA.

From Theorem 12, we have:

Corollary 4. *A language $L \subseteq w_1^* \cdots w_n^*$ is accepted by PDA with reversal-bounded counters if and only if it is accepted by a resetting PDA, where the machine makes no more than $2n-3$ stack reversals between resets.*

Proof. Let $L \subseteq w_1^* \cdots w_n^*$ is accepted by PDA with reversal-bounded counters. Then, by Theorem 12, $L = \cap_{j=1}^m L_j$ for some languages $L_1, \ldots, L_m$ such that each L_j can be accepted by $2n-3$ reversal-bounded PDA M_j. We can design a resetting PDA M that simulates M_j during the j-th pass and accepts the input if and only if all M_j's accept the input. Converse is also easy to show. □

6. Multitape PDAs with reversal-bounded counters

It is well-known that any unary language accepted by a PDA (i.e., unary CFL) is regular (i.e., accepted by an NFA). In this section, we generalize this result.

A set of strings is *1-bounded* if it is a subset of w^* for some nonnull string w. In the following, we generalize the notion of a language to a collection of n-tuples of strings. More precisely, we define a language L as $L \subseteq w_1^* \times \cdots \times w_n^*$. Such an n-tuple language can be recognized by an n-tape automaton M in which each of the strings w_j is stored on a read-only input tape with a right end marker. The input $(w_1, w_2, \ldots, w_n) \in L(M)$ if there is a sequence of moves starting from the initial configuration (i.e., the machine is in the initial state, each input head is on the left-end of its input tape with end marker, and if there is a stack, it contains only the bottom of the stack symbol) leading to acceptance (i.e., the state is accepting with all input heads on the end marker).

Theorem 13. *Let $L \subseteq w_1^* \times \cdots \times w_n^*$ be a set of n-tuples accepted by an n-tape PDA with reversal bounded counters, where $w_1, \ldots, w_n$ are nonnull strings. (Thus, each input tape is over a 1-bounded set). Then L can be accepted by a n-tape NFA.*

Proof. Let $a_1, \ldots, a_n$ be distinct symbols. We first construct a 1-tape PDA M_1 with reversal-bounded counters accepting $L_1 = \{a_1^{i_1} \cdots a_n^{i_n} \mid (w_1^{i_1}, \ldots, w_n^{i_n}) \in L\}$. M_1 has n new 1-reversal bounded counters, $C_1, \ldots, C_n$, in addition to the counters of M. M_1, when given input $a_1^{i_1} \cdots a_n^{i_n}$, reads the input and stores $i_1, \ldots, i_n$ in counters $C_1, \ldots, C_n$. Then M_1 simulates the computation of M by using $C_1, \ldots, C_n$ to simulate the movements of the n input heads of M: decrementing C_i corresponds to reading w_i. After reading an a_i, the computation on w_i is carried out in the finite-state control.

By Theorem 1, the set $Q_{L_1} = \{(i_1, \ldots, i_n) \mid a_1^{i_1} \cdots a_n^{i_n} \in L(M_1)\}$ is semilinear. Let $Q(c, V)$ be one of the linear sets comprising Q_{L_1}, where c in N^n is the constant vector, and $V = \{v_1, \ldots, v_r\}$, each v_i in N^n. Then $Q = \{(c_1, \ldots, c_n) + t_1(v_{11}, \ldots, v_{1n}) + \cdots + t_r(v_{r1}, \ldots, v_{rn}) \mid t_1, \ldots, t_n \in N\}$.

From Q, we construct an n-tape NFA M_2 accepting $L(Q) = \{(a_1^{i_1}, \ldots, a_n^{i_n}) \mid (i_1, \ldots, i_n) \in Q\}$. Given $(a_1^{i_1}, \ldots, a_n^{i_n})$, M_2 reads c_i a_i's on tape i. Then for $1 \le i \le r$, M_2 executes the following process $t_i \ge 0$ times (where t_i is nondeterministically chosen): Moves the input head j v_{ij} cells to the right. After the $t_r v_r$ has been processed, M_2 accepts. Clearly, M_2 accepts $L(Q)$. Since the languages accepted by n-tape NFAs are obviously closed under union, it follows that $L_3 = \{(a_1^{i_1}, \ldots, a_n^{i_n}) \mid (w_1^{i_1}, \ldots, w_n^{i_n}) \in L\}$ can be accepted by an n-tape NFA M_3.

From M_3, we can then construct another n-tape NFA M accepting L. □

Note that the above theorem does not hold when the tapes are no longer 1-bounded. For example, $L = \{(a^i b^i, a^j) \mid i, j \ge 1\}$ is accepted by a 2-tape PDA, but it cannot be accepted by any 2-tape NFA; otherwise, the projection of L on the first coordinate (which is not regular) could be accepted by a 1-tape NFA.

In the following, we consider a more general n-tuple language L in which each component language is a bounded (rather than a unary) language.

Let $B_1 \times \cdots \times B_n$ will denote a set of tuples, where each $B_i = w_{i1}^* \cdots w_{ik_i}^*$ for some nonnull strings $w_{i1}, \ldots, w_{ik_i}$. If $L \subseteq B_1 \times \cdots \times B_n$, define $Q_L = \{(i_{11}, \ldots, i_{1k_1}, \ldots, i_{n1}, \ldots, i_{nk_n}) \mid (w_{11}^{i_{11}} \cdots w_{1k_1}^{i_{1k_1}}, \ldots, w_{n1}^{i_{n1}} \cdots w_{nk_n}^{i_{nk_n}}) \in L\}$.

An n-tape automaton M is *serial* if it reads the input tapes in such a way that for $1 \le i < n$, M reads all the symbols in tape i before it reads tape $i+1$.

Theorem 14. *Let $L \subseteq B_1 \times \cdots \times B_n$ be accepted by an n-tape PDA with reversal-bounded counters. Then*

1. $L' = \{w_{11}^{i_{11}} \cdots w_{1k_1}^{i_{1k_1}} \cdots w_{n1}^{i_{n1}} \ldots w_{nk_n}^{i_{nk_n}} \mid (i_{11}, \ldots, i_{1k_1}, \ldots, i_{n1}, \ldots, i_{nk_n}) \in Q_L\}$ *can be accepted by a (1-tape) PDA with reversal-bounded counters.*
2. Q_L *is a semilinear set.*
3. L' *can be accepted by a DFA with reversal-bounded counters (note that the machine is deterministic and has no stack).*
4. *L can be accepted by a serial n-tape DFA with reversal-bounded counters.*

Proof. For part (1), we construct a PDA M', which when given input $w = w_{11}^{i_{11}} \cdots w_{1k_1}^{i_{1k_1}} \cdots w_{n1}^{i_{n1}} \cdots w_{nk_n}^{i_{nk_n}}$, reads the input and stores $i_{11}, \ldots, i_{1k_1}, \ldots, i_{n1}, \ldots, i_{nk_n}$ in new counters $C_{11}, \ldots, C_{1k_1}, \ldots, C_{n1}, \ldots, C_{nk_n}$. Then M' simulates the computation of M that accepts L by using these new counters: decrementing C_{ij} corresponds to reading w_{ij}. We omit the details. Part (2) follows from Theorem 1.

Part (3) follows from a result in [6], which showed that any language accepted by a PDA with reversal-bounded counters can be accepted by a DFA with reversal-bounded counters, i.e., the machine is deterministic and there is no stack.

Part (4) follows from part (3), since a DFA on input $w_{11}^{i_{11}} \cdots w_{1k_1}^{i_{1k_1}} \cdots w_{n1}^{i_{n1}} \cdots w_{nk_n}^{i_{nk_n}}$, can be simulated directly by an n-tape DFA with input $(w_{11}^{i_{11}} \cdots w_{1k_1}^{i_{1k_1}}, \ldots, w_{n1}^{i_{n1}} \cdots w_{nk_n}^{i_{nk_n}})$ where each tape has a right end marker. $\square$

Theorem 15. *Let $L \subseteq B_1 \times \cdots \times B_n$ be accepted by an n-tape PDA with reversal-bounded counters. If Q_L is a stratified semilinear set, then L can be accepted by a serial reversal-bounded n-tape PDA (thus, the stack is reversal-bounded and there are no counters).*

Proof. Let $a_{11}, \ldots, a_{1k_1}, \ldots, a_{n1}, \ldots, a_{nk_n}$ be distinct symbols, and let $L_1 = \{a_{11}^{i_{11}} \cdots a_{1k_1}^{i_{1k_1}} \cdots a_{n1}^{i_{n1}} \cdots a_{nk_n}^{i_{nk_n}} \mid (i_{11}, \ldots, i_{1k_1}, \ldots, i_{n1}, \ldots, i_{nk_n}) \in Q_L\}$. Then L_1 can be accepted by a reversal-bounded (1-tape) PDA M_1 by Theorem 4. Clearly, we can construct from M_1 a serial reversal-bounded n-tape PDA M' to accept $L' = \{(a_{11}^{i_{11}} \cdots a_{1k_1}^{i_{1k_1}}, \ldots, a_{n1}^{i_{n1}} \cdots a_{nk_n}^{i_{nk_n}}) \mid a_{11}^{i_{11}} \cdots a_{1k_1}^{i_{1k_1}} \cdots a_{n1}^{i_{n1}} \cdots a_{nk_n}^{i_{nk_n}} \in L_1\}$. From M', we can then construct a serial reversal-bounded n-tape PDA M'' accepting L. $\square$

As stated in Theorem 1, part 2, the Parikh map, $P(L)$, of a language L accepted by a PDA with reversal-bounded counters is a semilinear set. We will show that this generalizes to multitape machines.

Let $L \subseteq \Sigma_1^* \times \cdots \times \Sigma_n^*$. For $1 \le i \le n$, let $\Sigma_i = \{a_{i1}, \ldots, a_{ik_i}\}$. Define the Parikh map of L as $P(L) = \{(|w_1|_{a_{11}}, \ldots, |w_1|_{a_{1k_1}}, \ldots, |w_n|_{a_{n1}}, \ldots, |w_n|_{a_{nk_n}}) \mid (w_1, \ldots, w_n) \in L\}$.

Theorem 16. *If $L \subseteq \Sigma_1^* \times \cdots \times \Sigma_n^*$ is accepted by an n-tape PDA M with reversal-bounded counters, then P(L) is a semilinear set.*

Proof. We construct a PDA M' with reversal-bounded counters (note that M' has only one input tape) which accepts a language that is a subset of $a_{11}^* \cdots a_{1k_1}^* \cdots a_{n1}^* \cdots a_{nk_n}^*$ such that $P(L(M')) = P(L)$. M' has a stack and reversal-bounded counters for the simulation of M. In addition, M' has 1-reversal counters $c_{11}, \ldots, c_{1k_1}, \ldots, c_{n1}, \ldots, c_{nk_n}$, where c_{ij} is associated with symbol a_j in Σ_i.

M', when given an input $a_{11}^{t_{11}} \cdots a_{1k_1}^{t_{1k_1}} \cdots a_{n1}^{t_{n1}} \cdots a_{nk_n}^{t_{nk_n}}$, simulates the computation of M' (without reading the input) by guessing the symbols read by the n input heads of M on their respective tapes; each time head i has to read the next symbol, M' guesses a symbol a_{ij} as the symbol read by head i and increments counter c_{ij}. When M accepts, M' verifies that the value in counter c_{ij} is equal to t_{ij} for $1 \le i \le n$, $1 \le j \le k_i$, and then accepts. Clearly, $P(L(M')) = P(L)$. From Theorem 1, part 1, $P(L(M'))$ (and, hence, $P(L)$) is semilinear. $\square$

Suppose there are r distinct symbols in $\Sigma_1 \cup \cdots \cup \Sigma_n$: $a_1, \cdots, a_r$. Define $P(L)$ as $P(L) = \{(i_1, \ldots, i_r) \mid i_j = |w_1 \cdots w_n|_{a_j}, (w_1, \ldots, w_n) \in L, 1 \le j \le r\}$. Clearly by a construction similar to the proof above, $P(L)$ is also a semilinear set.

An n-tape 2PDA is a PDA with n two-way input tapes with left and right end markers on each tape. An n-tuple $(x_1, \ldots, x_n)$ is accepted if the machine, when started with its n input heads at the left end of their respective input tapes, eventually enters an accepting state with all input heads at the right end of their respective tapes. When there is no stack, we have an n-tape 2NFA. In the deterministic versions we use 'D' instead of 'N'. These models can be augmented with reversal-bounded counters.

An n-tape machine (of a given type) is *finite-turn* if there is a nonnegative integer c such that for any accepted n-tuple, there is an accepting computation in which on any input tape, the head makes at most c turns (from left-to-right or right-to-left). Note that if $c = 0$, the machine has one-way input tapes.

Proposition 2. *There is a language L accepted by a 2-turn 2DPDA whose Parikh map, $P(L)$, is not semilinear.*

Proof. Let $a, b, \#$ be new symbols, and $L = \{a^1ba^5 \cdots ba^{2n-1}\#a^{2n+1}b \cdots ba^7ba^3 \mid n \ge 1\}$, Clearly, L can be accepted by a 2-turn 2DPDA whose stack makes 3 reversals, but $P(L)$ is not semilinear. $\square$

However, for finite-turn 2NFAs, the following was shown in [5].

Theorem 17. *If $L \subseteq \Sigma^*$ is accepted by a finite-turn 2NFA with reversal-bounded counters, then $P(L)$ is semilinear.*

The above theorem does not hold for multitape machines, as the next proposition shows.

Proposition 3. *There is a language L accepted by a 2-turn 2-tape 2DFA such that $P(L)$ is not semilinear.*

Proof. Let $L = \{(a^1b^3 \cdots a^{n-1}, c^2d^4 \cdots c^{n-2}d^n) \mid n = 2i, i \geq 1\}$, where a, b, c, d are distinct symbols. Clearly, L can be accepted by a 2-turn 2-tape DFA, but $P(L)$ is not semilinear, since the projection of $P(L)$ on the first coordinate is the set of squares (and semilinear sets are closed under projections). □

In contrast to Proposition 2, for bounded languages, we have:

Theorem 18. *Let $L \subseteq a_{11}^* \cdots a_{1k_1}^* \times \cdots \times a_{n1}^* \cdots a_{nk_n}^*$ (where the a_{ij}'s are distinct symbols) be accepted by a finite-turn n-tape PDA M with reversal-bounded counters. Then P(L) is a semilinear set.*

Proof. We give the construction for $n = 2$, which easily generalizes. Let M be a 2-tape PDA with reversal-bounded counters accepting $L \subseteq a_1^* \cdots a_r^* \times b_1^* \cdots b_s^*$. We will show that $P(L) = \{(i_1, \ldots, i_r, j_1, \ldots, j_s) \mid (a_1^{i_1} \cdots a_r^{i_r}, b_1^{j_1} \cdots a_s^{j_s}) \in L\}$ is semilinear.

Construct a PDA M' which, in addition to the stack and reversal-bounded counters for simulating M, has new counters c_{i1} and c_{i2} associated with a_i for $1 \leq i \leq r$ and counters d_{j1} and d_{j2} associated with b_j for $1 \leq j \leq s$. M' will accept the language $L' = \{a_1^{i_1} \cdots a_r^{i_r} b_1^{j_1} \cdots b_s^{j_s} \mid (a_1^{i_1} \cdots a_r^{i_r}, b_1^{j_1} \cdots a_s^{j_s}) \in L\}$. M' operates as follows given input $a_1^{i_1} \cdots a_r^{i_r} b_1^{j_1} \cdots b_s^{j_s}$ on its tape.

M' first reads the input and stores $i_1, \ldots, i_r, j_1, \ldots, j_s$ in counters $c_{11}, \ldots, c_{r1}, d_{11}, \ldots, d_{s1}$, respectively. All other counters are zero initially. M' then simulates the computation of M by using counters c_{i1} and c_{i2} to track the position of the input head on tape 1 when it is on the a_i segment. Moving the head would correspond to incrementing one of these counters by one and decrementing the other by one. Similarly, counters d_{j1} and d_{j2} are used to track the position of the input head on tape 2 when it is on the b_j segment. Again, moving the head would correspond to incrementing one of these counters by one and decrementing the other by one. Since M is finite-turn, counters c_{i1}, c_{i2}, d_{j1}, d_{j2} will be reversal-bounded for all i and j. It follows from Theorem 1, part 1, that $P(L(M'))$ (and, hence, $P(L)$) is semilinear. □

Theorem 19. *Let $L \subseteq B_1 \times \cdots \times B_n$ be accepted by a finite-turn n-tape PDA with reversal-bounded counters. Then L can be accepted by:*

1. *A finite-turn n-tape 2DFA with one reversal-bounded counter.*
2. *An n-tape (one-way) DFA with a finite number of reversal-bounded counters.*

Proof. We prove the case $n = 2$, the generalization is straightforward. Let M be a 2-tape PDA with reversal-bounded counters accepting $L \subseteq w_1^* \cdots w_r^* \times x_1^* \cdots x_s^*$. Let $a_1, \ldots, a_r, b_1, \ldots, b_s$ be distinct symbols, and let $L_1 = \{(a_1^{i_1} \cdots a_r^{i_r}, b_1^{j_1} \cdots b_s^{j_s}) \mid (w_1^{i_1} \cdots w_r^{i_r}, x_1^{j_1} \cdots x_s^{j_s}) \in L\}$. It is easy to construct a 2-tape PDA M_1 with reversal-bounded counters accepting L_1. Then, as in the proof of Theorem 18, we can construct a PDA M_1' with reversal-bounded counters accepting $L_1' = \{a_1^{i_1} \cdots a_r^{i_r} b_1^{j_1} \cdots b_s^{j_s} \mid (a_1^{i_1} \cdots a_r^{i_r}, b_1^{j_1} \cdots b_s^{j_s}) \in L\}$. Hence, $P(L_1')$ is semilinear. It follows that the language $L_2 = \{w_1^{i_1} \cdots w_r^{i_r} x_1^{j_1} \cdots x_s^{j_s} \mid (w_1^{i_1} \cdots w_r^{i_r}, x_1^{j_1} \cdots x_s^{j_s}) \in L\}$ is a semilinear language.

In [6], it was shown that any semilinear language, like L_2, can be accepted by a finite-turn 2DFA with one reversal-bounded counter (and also by a one-way DFA with a finite number of reversal-bounded counters) M_2. Then from M_2 it is trivial to construct a finite-turn 2-tape DFA with one reversal-bounded counter (and a 2-tape one-way DFA with finite number of counters) accepting L. □

7. Conclusion

We conclude with the following open problems:

1. Show that L_{cycle}^n cannot be accepted by a counter machine for odd values of n.
2. Show that L_2^3 and L_3^3 are not deterministic context-free languages.
3. Show that L_2^n and L_3^n are inherently ambiguous for $n > 5$.
4. Show that the number of reversals needed to accept L_{cycle}^n is $(2n - 3)$ for all n except $n = 4$. While it seems unusual for the optimum bound to exhibit an exception like $n = 4$, Proposition 1 makes this conjecture plausible.

Acknowledgments

We thank the referees for helpful comments – especially referee 1 for a very careful reading and suggestions for improvements.

References

[1] R. Boonyavatana, G. Slutzki, A generalized Ogden's lemma for linear context-free languages, Bull. Eur. Assoc. Theor. Comput. Sci. 42 (1) (1985) 20–28.
[2] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages and Computation, Addison–Wesley Publishing Company, Massachusetts, 1979.
[3] S. Ginsburg, The Mathematical Theory of Context-Free Languages, McGraw–Hill Book Company, New York, 1966.
[4] S. Greibach, Linearity is polynomially decidable for real-time pushdown store automata, Inf. Control 42 (1) (1979) 27–37.
[5] O.H. Ibarra, Reversal-bounded multicounter machines and their decision problems, J. Assoc. Comput. Mach. 25 (1978) 116–133.
[6] O.H. Ibarra, S. Seki, Characterizations of bounded semilinear languages by one-way and two-way deterministic machines, Int. J. Found. Comput. Sci. 23 (2012) 1291–1306.
[7] M. Kutrib, A. Malcher, Finite turns and the regular closure of linear context-free languages, Discrete Appl. Math. 155 (2007) (2007) 2152–2164.
[8] L.Y. Liu, P. Weiner, A Characterization of semilinear sets, J. Comput. Syst. Sci. 4 (1970) 299–307.
[9] A. Malcher, G. Pighizzini, Descriptional complexity of bounded context-free languages, in: Proc. of the 11th International Conference on Developments in Language Theory, DLT 2007, 2007, pp. 313–323.
[10] R.J. Parikh, On context-free languages, J. Assoc. Comput. Mach. 13 (1966) 570–581.