Theoretical Computer Science 501 (2013) 11–19

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

On the open problem of Ginsburg concerning semilinear sets and related problems

Oscar H. Ibarra [a,*,1], Shinnosuke Seki [b,c,2]

[a] Department of Computer Science, University of California, Santa Barbara, CA 93106, USA
[b] Helsinki Institute for Information Technology (HIIT), Finland
[c] Department of Information and Computer Science, Aalto University, P.O. Box 15400, FI-00076, Aalto, Finland

ARTICLE INFO

Article history:
Received 11 April 2013
Received in revised form 7 July 2013
Accepted 18 July 2013
Communicated by D.-Z. Du

Keywords:
Multitape automata
Synchronizability
Semilinear set
1-reversal counters
Reversal-bounded counters
Stratified
Decidability

ABSTRACT

In his 1966 book, "The Mathematical Theory of Context-Free Languages", S. Ginsburg posed the following open problem: Find a decision procedure for determining if an arbitrary semilinear set is a finite union of stratified linear sets. It turns out that this problem (which remains open) is equivalent to the problem of synchronizability of multitape machines. Given an n-tape automaton M of a given type (e.g., nondeterministic pushdown automaton (NPDA), nondeterministic finite automaton (NFA), etc.) with a one-way read-only head per tape and a right end marker on each tape, we say that M is 0-synchronized (or aligned) if for every n-tuple $x = (x_1, \ldots, x_n)$ that is accepted, there is an accepting computation on x such that at any time during the computation, all heads except those that have reached the end marker are on the same position (i.e., aligned). One of our main contributions is to show that Ginsburg's problem is equivalent to deciding for an arbitrary n-tape NFA M accepting $L(M) \subseteq a_1^* \times \cdots \times a_n^*$ (where $n \geqslant 1$ and $a_1, \ldots, a_n$ are distinct symbols) whether there exists an equivalent 0-synchronized n-tape NPDA M'. Ginsburg's problem is decidable if M' is required to be an NFA, and we will generalize this decidability over bounded inputs. It is known that if the inputs are unrestricted, the problem is undecidable. We also show several other related decidability and undecidability results.
It may appear, as one of the referees suggested in an earlier version of this paper, that our main result can be written in first order logic. We explain why this is not the case in the Introduction.

1. Introduction

Motivated by applications to verification problems in string manipulating program (see, e.g., [10] for discussions on the need to validate input strings to avoid security vulnerabilities such as SQL injection attack), recent papers have looked at the problem of synchronizability of the input heads of multitape nondeterministic finite automata (NFAs), nondeterministic pushdown automata (NPDAs), etc. [2,6,7].

It turns out that questions concerning synchronization and synchronizability of multitape automata are intimately related to questions concerning properties of semilinear sets. We investigate these relationships in this paper. In particular, we look at the following open problem of S. Ginsburg in his 1966 book, "The Mathematical Theory of Context-Free Languages" [3]:

* Corresponding author.
E-mail addresses: ibarra@cs.ucsb.edu (O.H. Ibarra), shinnosuke.seki@aalto.fi (S. Seki).
1 Supported in part by NSF Grants CCF-1143892 and CCF-1117708.
2 Supported in part by HIIT Pump Priming Project Grants 902184/T30606.

0304-3975/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.tcs.2013.07.025

Find a decision procedure for determining if a given semilinear set is stratified. This is the same as asking, given a bounded semilinear language $L \subseteq a_1^* \cdots a_n^*$ (i.e., the set $\{(i_1, \ldots, i_n) \mid a_1^{i_1} \cdots a_n^{i_n} \in L\}$ is semilinear), is L context-free (i.e., accepted by an NPDA)? Interestingly, this problem (which remains open) is equivalent to the problem of synchronizability of multitape machines.

Given an n-tape automaton M of a given type (e.g., NPDA, NFA, etc.), with a one-way read-only head per tape and a right end marker \$ on each tape, we say that M is 0-synchronized (or aligned) if for every n-tuple $x = (x_1, \ldots, x_n)$ that is accepted, there is a computation on x such that at any time during the computation, all heads, except those that have reached the end marker, are on the same position (i.e., aligned). When a head reaches the marker, it can no longer move. As usual, an n-tuple $x = (x_1, \ldots, x_n)$ is accepted if M eventually reaches the configuration where all n heads are on \$ in an accepting state.

We will show that Ginsburg's problem is equivalent to deciding for an arbitrary n-tape NFA M accepting $L(M) \subseteq a_1^* \times \cdots \times a_n^*$ (where $n \geqslant 1$ and $a_1, \ldots, a_n$ are distinct symbols) whether there exists an equivalent 0-synchronized n-tape NPDA M'. The proof of this result also applies to the following special cases:

1. The following two statements are equivalent:
   - It is decidable whether a semilinear language $\subseteq a_1^* \cdots a_n^*$ is linear context-free;
   - It is decidable for an n-tape NFA over $a_1^* \times \cdots \times a_n^*$ whether there exists an equivalent 0-synchronized n-tape 1-turn NPDA (i.e., when the stack pops, it can no longer push).
2. The following two statements are equivalent:
   - It is decidable whether a semilinear language $\subseteq a_1^* \cdots a_n^*$ is regular;
   - It is decidable for an n-tape NFA over $a_1^* \times \cdots \times a_n^*$ whether there exists an equivalent 0-synchronized n-tape NFA.

The decidability of the first item is still open. Since the regularity test for semilinear sets is decidable [4], it follows from item 2 that synchronizability of an n-tape NFA is decidable. We will generalize this as: it is decidable, given an n-tape NFA M whose inputs come from $B_1 \times \cdots \times B_n$ (where each $B_i \subseteq w_1^* \cdots w_k^*$ for some $k \geqslant 1$ and nonempty words $w_1, \ldots, w_k$), whether there exists an equivalent 0-synchronized n-tape NFA M'. It is known that if the inputs are unrestricted, the problem is undecidable [2]. We also show several other related decidability and undecidability results.

A referee of an earlier version of this paper made the following comment: *The main result can be translated in terms of first order logic as follows: given a first order formula in Presburger arithmetic (i.e., semilinear set), decide whether or not it can be defined in Peladeau's logic (i.e., synchronized multitape automaton) [9]. Once translated, it suffices to apply Choffrut's decision procedure in [1].*

However, the suggestion that our main result could be obtained in terms of first order logic is not quite correct. Our understanding from the description above is that it is decidable to determine, given an n-tape NFA M accepting $L(M) \subseteq a_1^* \times \cdots \times a_n^*$ (i.e., a semilinear set over $\mathbb{N}^n$), whether there exists an equivalent 0-synchronized n-tape NFA M'. The referee attributed this as our main result, which is clearly not the case. Our main result is that Ginsburg's problem is equivalent to the question of whether or not for an n-tape NFA M there exists an equivalent 0-synchronized n-tape NPDA M' (which is still open).

The result that the referee suggested, in fact, as we already noted, follows from a result in [4] and the equivalence in item 2 above. Moreover, this decidability result does not imply the equivalences in items 1 and 2 above.

2. Preliminaries

For the set $\mathbb{N}$ of natural numbers and $n \geqslant 1$, $\mathbb{N}^n$ denotes the set of (n-dimensional nonnegative integer) vectors including the zero vector $\bar{0} = (0, 0, \ldots, 0)$. For a vector $v = (i_1, \ldots, i_n) \in \mathbb{N}^n$, $v[j]$ denotes its j-th component, that is, $v[j] = i_j$. A set of vectors $Q \subseteq \mathbb{N}^n$ is a *linear set* if there is a vector $c \in \mathbb{N}^n$ (constant vector) and a finite (possibly-empty) set of nonzero periodic vectors $V = \{v_1, \ldots, v_r\} \subseteq \mathbb{N}^n \setminus \{\bar{0}\}$ such that $Q = \{c + i_1 v_1 + \cdots + i_r v_r \mid i_1, \ldots, i_r \in \mathbb{N}\}$. We call the pair (c, V) a *linear generator* of Q. A set of vectors is a *semilinear set* if it is a finite union of linear sets. The set of linear generators of linear sets comprising Q is called a *generator* of Q.

Definition 1. A linear generator (c, V) 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 $1 \leqslant i < j < k < l \leqslant n$ and no vectors $u, v \in V$ such that none of $u[i]$, $v[j]$, $u[k]$, $v[l]$ is zero.

A linear set is *stratified* if it can be generated by a stratified linear generator. A finite union of stratified linear sets is called a *stratified semilinear set*.

For an alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$, let Σ^* be the set of words over Σ including the empty word ε and let $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. For a word $w \in \Sigma^*$, we denote by $|w|$ the number of letters (symbols) in w, and by $|w|_{a_i}$ the number of occurrences of a letter a_i in w. The *Parikh map* (or image) $\psi(w)$ of w is the vector $(|w|_{a_1}, \ldots, |w|_{a_n})$. The Parikh map of a language $L \subseteq \Sigma^*$ is defined as $\psi(L) = \{\psi(w) \mid w \in L\}$. A language $L \subseteq \Sigma^*$ is *bounded* if it is a subset of $w_1^* \cdots w_k^*$ for some $k \geqslant 0$ and nonempty words $w_1, \ldots, w_k \in \Sigma^+$. If all of $w_1, \ldots, w_k$ are pairwise-distinct letters, then L is especially called *letter-bounded*. A bounded language $L \subseteq w_1^* \cdots w_k^*$ is *semilinear* if the set $Q(L) = \{(i_1, \ldots, i_k) \mid w_1^{i_1} \cdots w_k^{i_k} \in L\}$ is semilinear.

We assume that the readers are familiar with nondeterministic finite automata (NFA) and pushdown automata. When we say that two machines are equivalent, we mean that they accept the same language. Nondeterministic counter machines (NCM) and pushdown counter machines (NPCM) are obtained by augmenting NFAs and NPDAs with counters (i.e., unary stacks) that are reversal-bounded [5]. A counter is *reversal-bounded* if the number of alternations between nondecreasing mode and nonincreasing mode during a computation is bounded by some nonnegative integer. In particular, a *1-reversal* counter can no longer increment once it decrements. It is easy to show that an r-reversal counter can be simulated by $\lceil (r+1)/2 \rceil$ 1-reversal counters [5]. In order to specify how many 1-reversal counters a machine has, we add an argument to its notation as NCM(m) (nondeterministic counter machine with m 1-reversal counters). Aside from increment by 1 and decrement by 1, the sole operation that can be carried out on counters is zero check.

It is known that if a language is accepted by an NPDA, then its Parikh map is an effectively computable semilinear set [8]. This was generalized in [5] for NPCMs as follows.

Theorem 1. *(See [5].) The following statements hold*:

1. *If $L \subseteq \Sigma^*$ is accepted by an NPCM, then $\psi(L)$ is an effectively computable semilinear set.*
2. *If a bounded language $L \subseteq w_1^* \cdots w_n^*$ is accepted by an NPCM, 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.*
3. *If $Q \subseteq \mathbb{N}^n$ is a semilinear set, then for any nonempty words $w_1, \ldots, w_n$, an NCM M can be constructed such that $L(M) \subseteq w_1^* \cdots w_k^*$ and $Q(L(M)) = Q$.*

Clearly, the class of bounded context-free languages is a subset of the class of bounded semilinear languages. The letter-bounded context-free languages are characterized in terms of Parikh mapping as follows.

Theorem 2. *(See [3].) A letter-bounded language $L \subseteq a_1^* \cdots a_n^*$ is context-free if and only if its Parikh map $\psi(L) \subseteq \mathbb{N}^n$ is a stratified semilinear set.*

3. Synchronized machines and synchronizability

For $k \geqslant 0$, a multitape machine (with a right end marker on each tape) is *k-synchronized* if, for any tuple of words it accepts, there exists a computation during which the distance between any pair of heads that have not reached the end marker is at most k. A multitape machine is *synchronized* if it is k-synchronized for some k.

For a machine class $\mathcal{M}$, any synchronized machine M in $\mathcal{M}$ admits an equivalent 0-synchronized machine in the same class. This follows from the observation that if M is k-synchronized, we can modify it to keep track of the contents of the tapes within the "window" of length k using "finite buffer" in the state control. In order for this conversion to be algorithmically feasible, it is crucial that the k must be explicitly given with M (see Corollary 2). Indeed, merely knowing that M is synchronized (i.e., k-synchronized for some k) only shows the existence of an equivalent 0-synchronized machine, but as we will see there is no algorithm for actually constructing this equivalent machine. This issue will be handled in Section 6.

0-synchronized n-tape machines are of special significance. This is because they can be considered rather working on one tape over an extended alphabet Π of *n-track symbols*, which is defined as:

$$\Pi = \left\{ \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \middle| a_1, \ldots, a_n \in \Sigma \cup \{\lambda\} \right\},$$

where $\lambda \notin \Sigma$ represents the blank symbol. Track symbols are distinguished from tuples of letters by square brackets, and for the space sake, written rather as $t = [a_1, \ldots, a_n]^T$; furthermore, the superscript T will be omitted unless confusion arises. For an index $1 \leqslant i \leqslant n$, $t[i]$ denotes the letter on the i-th track of t, that is, a_i.

We endow the track alphabet Π with the partial order $\preccurlyeq$. The order is defined as: $t_1 \preccurlyeq t_2$ for track symbols $t_1, t_2 \in \Pi$ if $t_1[i] = \lambda$ implies $t_2[i] = \lambda$ for all $1 \leqslant i \leqslant n$. For example, $[\lambda, b, c]$ is smaller than $[\lambda, \lambda, c]$ according to this order but incomparable with $[a, \lambda, c]$. An n-track word $t_1 t_2 \cdots t_m$ is *left-aligned* if $t_1 \preccurlyeq t_2 \preccurlyeq \cdots \preccurlyeq t_m$ holds. Informally speaking, on a left-aligned track word, once we find λ on a track, then to its right will be found nothing but λ's on the track. The left-aligned n-track word $t_1 t_2 \cdots t_m$ can be converted into the n-tuple $(h(t_1[1]t_2[1] \cdots t_m[1]), \ldots, h(t_1[n]t_2[n] \cdots t_m[n]))$ of words over Σ using the blank-symbol-erasing homomorphism $h : (\Sigma \cup \{\lambda\})^* \to \Sigma^*$, which is defined as $h(\lambda) = \varepsilon$ and $h(a) = a$ for any $a \in \Sigma$. For instance, $[a, b, c][\lambda, b, c][\lambda, \lambda, c]$ is thus converted into $(h(a\lambda\lambda), h(bb\lambda), h(ccc))$, which is (a, bb, ccc). By reversing this process, we can retrieve the original left-aligned n-track word from the resulting n-tuple of words. This one-to-one correspondence enables us to assume that 0-synchronized n-tape machines accept only left-aligned inputs.

For machine classes $\mathcal{M}_1$, $\mathcal{M}_2$, an n-tape machine $M \in \mathcal{M}_1$ is *synchronizable within* $\mathcal{M}_2$ if there exists a 0-synchronized n-tape machine $M' \in \mathcal{M}_2$ that is equivalent to M. Hence, $\mathcal{M}_2$ must be specified unless $\mathcal{M}_1 = \mathcal{M}_2$, which should be clear from the context.

Example 1. It is easy to construct a 2-tape NFA M that accepts $L = \{(a^n, (bb)^n) \mid n \geqslant 0\}$; each time the first head moves to the right, the second head moves twice, but this machine is not k-synchronized for any k. In fact, M is not synchronizable within the class of NFAs [7]. With the help of a 1-reversal counter, however, we can have a machine accept L while keeping its heads aligned. Hence, M is synchronizable within the class of NCM(1)s.

4. Ginsburg's open problem and synchronizability

In this section, we prove the equivalence between the decidability of context-freeness of a given bounded semilinear language and the decidability of synchronizability of a given multitape NFA over $a_1^* \times a_2^* \times \cdots \times a_n^*$ within the class of multitape NPDAs (Theorem 4), where $a_1, a_2, \ldots, a_n$ are pairwise-distinct letters. Ginsburg's open problem is thus rephrased in terms of head-synchronizability test. This result and its proof will be applied in order to prove the decidability of an analogous synchronizability test within the class of NFAs instead of within that of NPDAs, and its generalizations in Section 5.

It is known that semilinear sets are precisely the sets which are first order definable in Presburger arithmetic, and these sets are equivalent to rational sets which are sets definable by multitape automata over the direct product of integers [3]. Hence:

Theorem 3. *A set $Q \subseteq \mathbb{N}^n$ is semilinear if and only if the set $\{(a_1^{i_1}, \ldots, a_n^{i_n}) \mid (i_1, \ldots, i_n) \in Q\}$ can be accepted by an n-tape NFA. The construction is effective in both directions.*

Lemma 1. *For a letter-bounded semilinear language $L \subseteq a_1^* a_2^* \cdots a_n^*$, the language $L' = \{a_1^{i_1} a_2^{i_1+i_2} \cdots a_n^{i_1+i_2+\cdots+i_n} \mid a_1^{i_1} a_2^{i_2} \cdots a_n^{i_n} \in L\}$ is also letter-bounded semilinear.*

Proof. Being semilinear, L can be accepted by an NCM M [5]. From M, we construct another NCM M' that computes $i_1, i_2, \ldots, i_n$, and stores them in additional counters while scanning an input $a_1^{i_1} a_2^{i_1+i_2} \cdots a_n^{i_1+i_2+\cdots+i_n}$. Using the stored counts $i_1, \ldots, i_n$, M' then simulates the computation of M on the input. It follows now from Theorem 1 that L' is a semilinear language. $\square$

Theorem 4. *The following statements are equivalent*:

1. *It is decidable, given a semilinear language $L \subseteq a_1^* \cdots a_n^*$, whether it is context-free (i.e., accepted by an NPDA).*
2. *It is decidable, given an n-tape NFA M over $a_1^* \times \cdots \times a_n^*$, whether there exists a 0-synchronized n-tape NPDA M' such that $L(M') = L(M)$.*

Proof. ((1) $\Rightarrow$ (2)) Let M be an n-tape NFA over $a_1^* \times \cdots \times a_n^*$. Let $\alpha_1, \ldots, \alpha_{2^n} \in \Pi$ be an ordered list of n-track symbols, where $\alpha_1 = [a_1, \ldots, a_n]$, $\alpha_{2^n} = [\lambda, \ldots, \lambda]$, and $\alpha_2, \ldots, \alpha_{2^n-1}$ are listed in the following order: all n-track symbols with exactly 1 track equal to λ are listed first, then all n-track symbols with exactly 2 tracks equal to λ are listed, and so on. Thus, for example, $\alpha_2 = [\lambda, a_2, \ldots, a_n]$, $\alpha_3 = [a_1, \lambda, a_3, \ldots, a_n]$, $\ldots$, $\alpha_{n+1} = [a_1, \ldots, a_{n-1}, \lambda]$, $\alpha_{n+2} = [\lambda, \lambda, a_3, \ldots, a_n]$, and so forth.

We construct a (1-tape) NCM M_1 that operates in the following manner: Given an n-track word $w \in \alpha_1^* \cdots \alpha_{2^n}^*$:

1. In scanning w, M_1 checks whether it is left-aligned. If not, w is rejected.
2. M_1 uses n 1-reversal counters $C_1, \ldots, C_n$, which are initially zero. M_1 increments C_i whenever it reads a track symbol whose i-th track is not λ. When M_1 sees the i-th track be λ, C_i will not increment any more.
3. After M_1 reads the whole input w, each counter C_i contains some value j_i (the number of track symbols on w whose i-th track is not λ). Then M_1 can simulate M's computation on $a_1^{j_1} \times \cdots \times a_n^{j_n}$ by using the counters $C_1, \ldots, C_n$. (M_1 decrements C_i to simulate reading a_i.)
4. M_1 accepts if M accepts.

Clearly, the counters $C_1, \ldots, C_n$ are 1-reversal. From Theorem 1, $L(M_1) \subseteq \alpha_1^* \cdots \alpha_{2^n}^*$ is a semilinear language. (Note that if w is in $L(M_1)$, then w is either the empty word or in $\alpha_{j_1}^+ \cdots \alpha_{j_r}^+$ for some $1 \leqslant r \leqslant n$ and $j_1 < j_2 < \cdots j_r < 2^n$.)

Using (1), we decide whether this semilinear language $L(M_1)$ is context-free or not. If the answer is positive, then there exists a 1-tape NPDA M_2 that accepts $L(M_1)$. It should be now straightforward that M_2 can be regarded rather as an n-tape 0-synchronized NPDA that accepts $L(M)$. In order to show that the negative answer implies the nonexistence of 0-synchronized NPDA for $L(M)$, we consider its contrapositive, and it is almost trivial because from a 0-synchronized NPDA that accepts $L(M)$, we can immediately build a 1-tape NPDA that accepts $L(M_1)$, and hence, $L(M_1)$ is context-free. Consequently, the decision problem (2) can be reduced to the decision problem (1).

((2) $\Rightarrow$ (1)) Let $L \subseteq a_1^* a_2^* \cdots a_n^*$ be a semilinear language. Then by Lemma 1, $L_1 = \{a_1^{i_1} a_2^{i_1+i_2} \cdots a_n^{i_1+i_2+\cdots+i_n} \mid a_1^{i_1} a_2^{i_2} \cdots a_n^{i_n} \in L\}$ is also semilinear. By Theorem 3, we can effectively construct an n-tape NFA M_3 that accepts the n-tuple language

$$L_2 = \{(a_1^{i_1}, a_2^{i_1+i_2}, \ldots, a_n^{i_1+i_2+\cdots+i_n}) \mid a_1^{i_1} a_2^{i_1+i_2} \cdots a_n^{i_1+i_2+\cdots+i_n} \in L_1\}.$$

We claim that M_3 admits an equivalent 0-synchronized n-tape NPDA if and only if L is context-free; (1) follows from (2) once this claim is confirmed. Assume that L_2 can be accepted by a 0-synchronized n-tape NPDA M_4. We consider inputs to M_4 as track words taken from $t_1^* t_2^* \cdots t_n^*$, where

$$t_1 = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix}, \quad t_2 = \begin{bmatrix} \lambda \\ a_2 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix}, \quad \ldots, \quad t_{n-1} = \begin{bmatrix} \lambda \\ \lambda \\ \vdots \\ \lambda \\ a_n \end{bmatrix}, \quad t_n = \begin{bmatrix} \lambda \\ \lambda \\ \vdots \\ \lambda \\ \lambda \end{bmatrix}.$$

Then M_4 accepts $t_1^{i_1} t_2^{i_2} \cdots t_n^{i_n}$ if and only if $a_1^{i_1} a_2^{i_2} \cdots a_n^{i_n}$ is in L. As a result, L is context-free.

For the converse implication, if L is context-free, then there is a (1-tape) NPDA M_5 accepting L. We can then directly construct from M_5 another (1-tape) NPDA M_6 (on the n-track symbols $t_1, \ldots, t_n$ defined above) accepting $t_1^{i_1} t_2^{i_2} \cdots t_n^{i_n}$ if and only if $a_1^{i_1} a_2^{i_2} \cdots a_n^{i_n}$ is accepted by M_5. It should be trivial now that M_6 can be regarded as a 0-synchronized n-tape NPDA, which is equivalent to M_3. $\square$

Solving the open problem by Ginsburg has turned out to be equivalent to determining the synchronizability of a given multitape NFA within the class of multitape NPDAs. Note that we can interpret this result also in terms of the stratifiability test of semilinear sets (Theorem 2).

The proof of Theorem 4 applies directly to the next result. The stack of an NPDA is *t-turn* if the number of alternations between pushing mode and popping mode during any computation is at most t times. Thus, for instance, a 1-turn NPDA can no longer push once it pops.

Theorem 5. *The following statements are equivalent*:

1. *It is decidable, given a semilinear language $L \subseteq a_1^* \cdots a_n^*$, whether it is linear context-free, i.e., accepted by a* 1*-turn NPDA* (*resp., regular, i.e., accepted by an NFA*).
2. *It is decidable, given an n-tape NFA M over $a_1^* \times \cdots \times a_n^*$, whether there exists a* 0*-synchronized n-tape* 1*-turn NPDA* (*resp.,* 0*-synchronized n-tape NFA*) *M' such that $L(M') = L(M)$.*

The above two theorems can be generalized further by augmenting the machines with 1-reversal counters.

Theorem 6. *For $k \geqslant 0$, the following statements are equivalent*:

1. *It is decidable, given a semilinear language $L \subseteq a_1^* \cdots a_n^*$, whether it is accepted by an* NCM(k) (*resp.,* 1*-turn* NPCM(k), NPCM(k)).
2. *It is decidable, given an n-tape NFA M over $a_1^* \times \cdots \times a_n^*$, whether there exists a* 0*-synchronized n-tape* NCM(k) (*resp.,* 0*-synchronized n-tape* 1*-turn* NPCM(k), 0*-synchronized n-tape* NPCM(k)) *M' such that $L(M') = L(M)$.*

Although the decidability of context-freeness of a semilinear language remains open, we can prove decidability for a restricted class. Assume that an NPDA can push or pop at most one symbol per move.

Proposition 1. *It is decidable, given a semilinear language $L \subseteq w_1^* \cdots w_k^*$ and $p, s \geqslant 1$, whether there is an NPDA M' with p pushdown stack symbols and s states that accepts L.*

Proof. Let M be an NFA augmented with reversal-bounded counters accepting L. We can systematically enumerate exhaustively all NPDAs M' with at most p pushdown symbols and s states (there are only a finite number of them) and check if they are equivalent to M. The result follows, since equivalence of NPDAs augmented with reversal-bounded counters is decidable [5]. $\square$

5. Synchronizability of *n*-tape NFAs over bounded tapes

According to Theorem 5, the problem of deciding the synchronizability of an n-tape NFA over $a_1^* \times \cdots \times a_n^*$ can be reduced to the regularity test of a semilinear language $\subseteq a_1^* \cdots a_n^*$. Since the latter is known to be decidable [4], we have:

Theorem 7. *It is decidable whether a given n-tape NFA M over $a_1^* \times \cdots \times a_n^*$ is synchronizable (i.e., whether there exists a 0-synchronized n-tape NFA M' equivalent to M).*

The equivalent 0-synchronized n-tape NFA M' in the statement of Theorem 7 is indeed effectively constructible. The algorithm is an exhaustive search; for each n-tape NFA M' generated exhaustively in the order of increasing the number of states, we check whether M' is both equivalent to M and 0-synchronized, both of which are testable based on the following two results (in their statements, each B_i is bounded, i.e., $B_i \subseteq w_1^* \cdots w_k^*$ for some $k \geqslant 1$ and nonempty words $w_1, \ldots, w_k$):

Lemma 2. *(See [5].) It is decidable, given two n-tape NPDAs M_1 and M_2 over a bounded input $B_1 \times \cdots \times B_n$, whether $L(M_1) = L(M_2)$.*

Lemma 3. *It is decidable, given an NFA M_1 over a bounded input $B_1 \times \cdots \times B_n$, whether it is 0-synchronized.*

Proof. We construct an n-tape NFA M_2 that simulates M_1 but makes sure that during the simulation, every input head that has not reached the end marker is on the same cell. Then decide whether $L(M_1) = L(M_2)$ (by Lemma 2). □

With this construction algorithm, Theorem 7 implies:

Corollary 1. *It is decidable whether a given n-tape NFA M over $a_1^* \times \cdots \times a_n^*$ is synchronizable. If M is synchronizable, then a 0-synchronized n-tape NFA that is equivalent to M can be effectively constructed.*

We will show that Corollary 1 generalizes to the case when the n-tape NFA is over $w_1^* \times \cdots \times w_n^*$, where the w_i's are nonempty words. This cannot be obtained simply by converting M into an n-tape NFA M' over $a_1^* \times \cdots \times a_n^*$ such that $L(M') = \{(a_1^{i_1}, \ldots, a_n^{i_n}) \mid (w_1^{i_1}, \ldots, w_n^{i_n}) \in L(M)\}$ and deciding the synchronizability of $L(M')$ using Corollary 1. This is because, in general, this conversion does not preserve the synchronizability of languages, as the example below shows.

Example 2. We have mentioned earlier that the language $\{(a^n, (bb)^n) \mid n \geqslant 0\}$ cannot be accepted by any 0-synchronized 2-tape NFA, and hence, not synchronizable within the class of NFAs. In contrast, it is obvious that $\{(a^n, b^n) \mid n \geqslant 0\}$ can be accepted by a 0-synchronized 2-tape NFA.

Lemma 4. *Let $\mathcal{M}$ be a class of machines. For nonempty words $w_1, \ldots, w_n$, let $L \subseteq w_1^* \times \cdots \times w_n^*$ be accepted by an n-tape machine $M \in \mathcal{M}$. For distinct letters $a_1, \ldots, a_n$ we can construct an n-tape machine $M' \in \mathcal{M}$ such that*

$$L(M') = \left\{\left(\left(a_1^{|w_1|}\right)^{j_1}, \ldots, \left(a_n^{|w_n|}\right)^{j_n}\right) \,\middle|\, \left(w_1^{j_1}, \ldots, w_n^{j_n}\right) \in L(M)\right\},$$

and M' is synchronizable within $\mathcal{M}$ if and only if so is M.

Proof. The words $w_1, \ldots, w_n$ are encoded in the states. M' simulates M in such a way that for every w_i read by the i-th head of M on tape i, the i-th head of M' reads $a_i^{|w_i|}$. Thus, if the i-th head of M has scanned $w_i^{j_i}$, the i-th head of M' has scanned $(a_i^{|w_i|})^{j_i}$.

One can verify the following: If there exists a 0-synchronized machine $M_1 \in \mathcal{M}$ accepting $L(M)$, then we can construct from M_1 a 0-synchronized machine $M_1' \in \mathcal{M}$ accepting $L(M')$, and conversely. □

Theorem 8. *It is decidable whether a given n-tape NFA M over $w_1^* \times \cdots \times w_n^*$ is synchronizable, where $w_1, \ldots, w_n$ are nonempty words. If M is synchronizable, a 0-synchronized n-tape NFA that is equivalent to M can be effectively constructed.*

We will generalize this theorem further using the following lemmas. The proof of the first lemma uses the same idea as in the proof of Lemma 4.

Lemma 5. *Let $\mathcal{M}$ be a class of machines. For $k \geqslant 1$ and nonempty words $w_1, \ldots, w_k$, let $L \subseteq w_1^* \cdots w_k^* \times \Sigma^* \times \cdots \times \Sigma^*$ be accepted by an n-tape machine $M \in \mathcal{M}$. For distinct letters $a_1, \ldots, a_k$, we can construct an n-tape machine $M' \in \mathcal{M}$ over $a_1^* \cdots a_k^* \times \Sigma^* \times \cdots \times \Sigma^*$ such that*

$$L(M') = \left\{\left(\left(a_1^{|w_1|}\right)^{j_1} \cdots \left(a_k^{|w_k|}\right)^{j_k}, z_2, \ldots, z_n\right) \,\middle|\, \left(w_1^{j_1} \cdots w_k^{j_k}, z_2, \ldots, z_n\right) \in L(M)\right\},$$

and M' is synchronizable within $\mathcal{M}$ if and only if so is M.

Lemma 6. *For $k \geqslant 1$ and distinct letters $a_1, a_2, \ldots, a_k$, let M be an n-tape NFA accepting $L \subseteq a_1^* a_2^* \cdots a_k^* \times \Sigma^* \times \cdots \times \Sigma^*$. We can construct a $(k+n-1)$-tape NFA M' accepting*

$$L' = \left\{\left(a_1^{i_1}, a_2^{i_1+i_2}, \ldots, a_k^{i_1+i_2+\cdots+i_k}, w_2, \ldots, w_n\right) \,\middle|\, \left(a_1^{i_1} a_2^{i_2} \cdots a_k^{i_k}, w_2, \ldots, w_n\right) \in L\right\}$$

such that M' is synchronizable if and only if so is M.

Proof. The first k heads of M' move synchronously while simulating the first head of M on the string $a_1^{i_1}a_2^{i_2}\cdots a_k^{i_k}$ on the first tape. The remaining $(n-1)$ heads of M' simulate the other $(n-1)$ heads of M on the 2nd to n-th tapes, respectively. It is easy to verify that there is a synchronized n-tape NFA M_1 equivalent to M if and only if there is a synchronized $(k+n-1)$-tape NFA M_1' equivalent to M'. □

Theorem 9. *It is decidable whether a given n-tape NFA M over bounded inputs $B_1 \times \cdots \times B_n$ is synchronizable. If M is synchronizable, then a 0-synchronized n-tape NFA that is equivalent to M can be effectively constructed.*

Proof. By applying Lemmas 5 and 6 iteratively, we can construct from M an s-tape NFA M' over $b_1^* \times \cdots \times b_s^*$ (for some $s \geqslant 1$ and distinct symbols $b_1, \ldots, b_s$) such that M' is synchronizable if and only if M is synchronizable. The decidability follows from Corollary 1. The construction of the equivalent 0-synchronized NFA employs the exhaustive-search algorithm introduced just after Theorem 7. □

6. Decidability and computability concerning head-synchronization

For $r \geqslant 1$, a machine is *r-ambiguous* if every input has at most r accepting computations. Thus, 1-ambiguous is the same as unambiguous. We first recall three recent results [2,6] (where "synchronized" was called "weakly synchronized").

Theorem 10. *(See [2].) The following problems are undecidable, given a 2-ambiguous 2-tape NFA M:*

1. *Is M k-synchronized for a given k?*
2. *Is M k-synchronized for some k?*
3. *Is there a 2-tape NFA M' that is 0-synchronized (or k-synchronized for a given k, or k-synchronized for some k) such that $L(M') = L(M)$?*

Recall the notion of t-turn stack for an NPDA, defined in Section 4.

Theorem 11. *(See [6].) Let Σ be an alphabet with at least two letters, and a be a letter. The following problems are undecidable, given a 2-ambiguous 2-tape 3-turn NPDA M over $\Sigma^* \times a^*$:*

1. *Is M k-synchronized for a given k?*
2. *Is M k-synchronized for some k?*
3. *Is there a 2-tape NPDA M' that is 0-synchronized (or k-synchronized for a given k, or k-synchronized for some k) such that $L(M') = L(M)$?*

We note that parts (1) and (2) of the above theorems have been shown to be decidable when the machine is unambiguous (i.e., 1-ambiguous) [2,6].

Theorem 12. *(See [2].) It is decidable whether a given unambiguous n-tape NFA is synchronizable, and if it is, we can effectively compute the smallest k_0 such that the machine is k_0-synchronized.*

In contrast, if we are given, along with a multitape NFA M, an integer $k \geqslant 0$ such that M is k-synchronized, then Algorithm 1 below enables us to effectively determine how far the heads of M are necessary apart, that is, we can compute the smallest integer k_0 such that M is k_0-synchronized. Note that k being explicitly given is crucial.

Algorithm 1 Given a multitape NFA M that is k-synchronized, find the minimum $k_0 \geqslant 0$ such that M is k_0-synchronized.

```
1: Convert M into an equivalent but 0-synchronized NFA M'.
2: for 0 ⩽ i ⩽ k do
3:     Construct an i-synchronized NFA M_i by equipping M with a mechanism to abort the computation as soon as two heads of M neither of which is on
       the end marker are separated by at least i + 1 cells.
4:     Convert M_i further into an equivalent 0-synchronized NFA M_i'.
5:     if L(M_i') = L(M') then
6:         return i
7:     end if
8: end for
```

It is the conversion in line 1 that needs the k such that M is k-synchronized be explicitly given.

These contrasting results give rise to the following problem: for a given multitape NFA M, does merely knowing that M is synchronized make it possible to decide whether M is 0-synchronized? The answer is no, as we show below.

Theorem 13. *It is undecidable whether a given synchronized 2-ambiguous 2-tape NFA is 0-synchronized.*

Proof. We reduce the undecidability of the halting problem for deterministic TMs on blank tape to our problem. Let Γ be the alphabet used to encode the sequence w of instantaneous descriptions (IDs), separated by the separators #'s, that describes the TM's accepting computation for the empty word, if the TM accepts it. Let $L = L_1 \cup L_2$, where $L_1 = \{(x, y) \mid x, y \in \Gamma^*, x \neq y\}$ and $L_2 = \{(w, w)\}$. Note that the TM accepts the empty word if and only if $L_2 \neq \emptyset$.

We design a 2-ambiguous 2-tape NFA M that accepts L. First it nondeterministically guesses whether a given input is in L_1 or L_2 as follows:

1. If the guess is for L_1, then it moves both heads to the right simultaneously and accepts if $x \neq y$. Clearly, for this computation, M is deterministic and 0-synchronized.
2. If the guess is for L_2, M assumes that the first tape and second tape are identical. (During the computation, if the tapes are different, then it does not matter what M does in this case, as the input is accepted with guess for L_1, anyway.) M initially moves its second head until it reads the first ID q_0# without moving the first head. M then checks that the (identical tapes) correspond to the computation of the TM. The first head of M would lag by one ID until the very end stage when the second head reaches an accepting state, after which the first head moves to the right end marker, and M accepts. Again the process is deterministic. The initialization guarantees that the lag is bigger than 0, but upperbounded by some constant.

Thus, M is synchronized and 2-ambiguous because after making the nondeterministic guess between L_1 and L_2, the remaining computation proceeds deterministically. It follows that M is 0-synchronized if and only if the TM does not accept the empty word. □

Corollary 2. *There is no algorithm to convert a synchronized 2-ambiguous 2-tape NFA M to an equivalent 0-synchronized 2-tape NFA.*

Proof. Suppose there were an algorithm to convert M to an equivalent 0-synchronized 2-tape NFA M'. We construct from M a 0-synchronized 2-tape NFA M'' which simulates M but only accepts when all heads are 0-synchronized during the computation. Then M is 0-synchronized if and only if $L(M'') = L(M')$, which is decidable. Hence, one could decide if M is 0-synchronized, which contradicts Theorem 13. □

Corollary 3. *There is no algorithm to compute, for a given synchronized 2-ambiguous 2-tape NFA M, an integer $k \geqslant 0$ such that M is k-synchronized.*

Proof. If we were able to compute k, then Algorithm 1 could minimize it. Contradictorily, this would amount to a decision procedure for the problem stated in Theorem 13. □

Finally, we give a positive result for multitape NFAs over bounded inputs:

Theorem 14. *Given a synchronized multitape NFA M over bounded inputs, the minimum integer $k_0 \geqslant 0$ such that M is k_0-synchronized is computable.*

Proof. For $i \geqslant 0$, we construct a multitape NFA M_i that simulates M and accepts a given input only if during the accepting computation, no two heads neither of which has reached the end marker are separated by more than i cells. Then k_0 is the minimum i such that $L(M_i) = L(M)$, which is decidable, since equivalence between multitape NFAs (in fact, even between multitape NPDAs) over bounded input tapes is decidable [5]. □

7. Conclusion

We proved that Ginsburg's open problem of determining if an arbitrary semilinear set is a finite union of stratified linear sets is equivalent to deciding for an arbitrary n-tape NPDA M accepting a bounded language if there exists a synchronized n-tape NPDA M' such that $L(M') = L(M)$. We also obtained related results concerning semilinear sets and synchronizability of multitape nondeterministic finite automata and counter machines. However, Ginsburg's problem remains open.

Acknowledgement

We thank the referees of an earlier version of this paper for their suggestions and comments that improve the presentation of our results.

References

[1] C. Choffrut, Deciding whether a relation defined in Presburger logic can be defined in weaker logics, RAIRO ITA 42 (1) (2008) 121–135.
[2] Ö. Eğecioğlu, O.H. Ibarra, N.Q. Tran, Multitape NFA: Weak synchronization of the input heads, in: SOFSEM 2012, in: LNCS, vol. 7147, Springer, 2012, pp. 238–250.

[3] S. Ginsburg, The Mathematical Theory of Context-Free Languages, McGraw–Hill, New York, 1966.
[4] S. Ginsburg, E. Spanier, Bounded regular sets, Proc. Am. Math. Soc. 17 (5) (1966) 1043–1049.
[5] O.H. Ibarra, Reversal-bounded multicounter machines and their decision problems, J. Assoc. Comput. Mach. 25 (1978) 116–133.
[6] O.H. Ibarra, N.Q. Tran, Weak synchronization and synchronizability of multitape pushdown automata and Turing machines, in: LATA 2012, in: LNCS, vol. 7183, Springer, 2012, pp. 337–350.
[7] O.H. Ibarra, N.Q. Tran, How to synchronize the heads of a multitape automaton, in: CIAA 2012, in: LNCS, vol. 7381, Springer, 2012, pp. 192–204.
[8] R.J. Parikh, On context-free languages, J. Assoc. Comput. Mach. 13 (1966) 570–581.
[9] P. Peladeau, Logically defined subsets of N^k, Theoret. Comput. Sci. 93 (2) (1992) 169–183.
[10] F. Yu, T. Bultan, O.H. Ibarra, Relational string verification using multi-track automata, Int. J. Found. Comput. Sci. 22 (2011) 1909–1924.