Multitape NFA: Weak Synchronization of the Input Heads

Ömer Eğecioğlu[1], Oscar H. Ibarra[1,⋆], and Nicholas Q. Tran[2]

[1] Department of Computer Science
University of California, Santa Barbara, CA 93106, USA
{omer,ibarra}@cs.ucsb.edu
[2] Department of Mathematics & Computer Science
Santa Clara University, Santa Clara, CA 95053, USA
ntran@math.scu.edu

Abstract. Given an n-tape nondeterministic finite automaton (NFA) M with a one-way read-only head per tape and a right end marker \$ on each tape, and a nonnegative integer k, we say that M is weakly k-synchronized 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, no pair of input heads, neither of which is on \$, are more than k cells apart. 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 show decidable and undecidable results concerning questions such as: (1) Given M, is it weakly k-synchronized for some k (resp., for a specified k) and (2) Given M, is there a weakly k-synchronized M' for some k (resp., for a specified k) such that $L(M') = L(M)$? Most of our results are the strongest possible in the sense that slight restrictions on the models make the undecidable problems decidable. A few questions remain open.

Keywords: multitape NFA, weakly synchronized, (un)decidability.

1 Introduction

Motivated by applications to verification problems in string manipulating program (see, e.g., [5, 6, 7] for discussions on the need to validate input strings to avoid security vulnerabilities such as SQL injection attack), we look at the problem of whether the input heads in a multitape nondeterministic finite automaton (NFA) are *weakly k-synchronized*, i.e., for each accepted input there is an accepting computation where no pair of inputs heads, neither of which is on \$, are more than k tape cells apart at any time.

In a recent paper [2], we studied a different notion of head synchronization: an n-tape NFA M is *strongly k-synchronized* if at any time during any computation on *any* input n-tuple $(x_1, \ldots, x_n)$ (accepted or not), no pair of input heads,

⋆ Supported in part by NSF Grants CCF-1143892 and CCF-1117708.

M. Bieliková et al. (Eds.): SOFSEM 2012, LNCS 7147, pp. 238–250, 2012.

neither of which is on \$, are more than k tape cells apart. In [2], we showed the following (among other things):

(**) It is decidable to determine, given an n-tape NFA M, whether it is k-synchronized for some k, and if this is the case, the smallest such k can be found.

Strong synchronization (studied in [2]) is a more restrictive requirement than what we investigate in this paper. Obviously, a strongly synchronized machine is also weakly synchronized, but the converse is not true. Consider, e.g., the set $L = \{(a^m\$, b^n\$) \mid m, n > 0\}$. We can construct a 2-tape NFA M, which when given input $(a^m\$, b^n\$)$, nondeterministically executes (1) or (2) below:

1. M reads $a^m\$$ on tape 1 until head 1 reaches \$, and then reads $b^n\$$ on tape 2 until head 2 reaches \$ and then accepts.
2. M reads the symbols on the two tapes simultaneously until one head reaches \$. Then the other head scans the remaining symbols on its tape and accepts.

Then M is not strongly synchronized, because of (1). However, M is weakly synchronized (in fact, weakly 0-synchronized) because every tuple $(a^m\$, b^n\$)$ can be accepted in a computation as described in (2). Thus strongly synchronized implies weakly synchronized, but not conversely.

It turns out that questions concerning weak synchronization are harder to answer than those for strong synchronization. Moreover, these two cases give some contrasting results. For example we show that, unlike (**) above, it is undecidable to determine, given a 2-ambiguous 2-tape NFA, whether it is weakly k-synchronized. However, the problem is decidable if M is 1-ambiguous, i.e., unambiguous. (A machine is k-ambiguous if there are at most k accepting computations for any input. Note that deterministic is a special case of 1-ambiguous.)

Note: Some proofs are omitted due to lack of space. All proofs will be given in a full version of the paper.

2 Preliminaries

An n-tape NFA M is a finite automaton with n tapes where each tape contains a string over input alphabet Σ. Each tape is read-only and has an associated one-way input head. We assume that each tape has a right end marker \$ (not in Σ). On a given n-tuple input $x = (x_1, \ldots, x_n)$, M starts in initial state q_0 with all the heads on the first symbols of their respective tapes. The transition function δ of M with state set Q is a mapping from $Q \times (\Sigma \cup \{\$\})^n \to 2^{Q \times \{0,1\}^n}$. If M is in state q with head H_i on symbol a_i and $(p, d_1, \ldots, d_n)$ is in $\delta(q, a_i, \ldots, a_n)$, then the machine moves H_i in direction d_i which is 1 or 0 (for right move or stationary move), and enters state p. When a head reaches the end marker \$, that head has to remain on the end marker. The input x is accepted if M eventually reaches the configuration where all n heads are on \$ in an accepting state.

Let M be an n-tape NFA and $k \geq 0$. M is *weakly k-synchronized* 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, no pair of input heads, neither of which is on \$, are more than k cells apart. Notice that, since the condition in the definition concerns pairs of heads that are both on symbols in Σ, if one of these two heads is on \$, then we can stipulate that the condition is automatically satisfied, irrespective of the distance between the heads. In particular, if $k = 0$, then all heads move to the right synchronously at the same time (except for heads that reach the right end marker early). M is *weakly synchronized* if it is weakly k-synchronized for some k.

An n-tape NFA that is deterministic is called an n-tape DFA. An n-tape NFA (DFA) can be augmented with a finite number of reversal-bounded counters. At each step, each counter (which is initially set to zero and can never become negative) can be incremented by 1, decremented by 1, or left unchanged and can be tested for zero. The counters are reversal-bounded in the sense that there is a specified r such that during any computation, no counter can change mode from increasing to decreasing and vice-versa more than r times. A counter is 1-reversal if once it decrements, it can no longer increment. Clearly, an r-reversal counter can be simulated by $\lceil (r+1)/2 \rceil$ 1-reversal counters.

Given an n-tuple $(x_1, \ldots, x_n)$, denote by $\langle x_1, \ldots, x_n \rangle$ an n-track string where the symbols of x_i's are left-justified (i.e., the symbols are aligned) and the shorter strings are right-filled with blanks (λ) to make all tracks the same length. For example, $\langle 01, 1111, 101 \rangle$ has $01\lambda\lambda$ on the upper track, 1111 on the middle track, and 101λ on the lower track. Given a set L of n-tuples, define $\langle L \rangle = \{\langle x \rangle \mid x \in L\}$.

Lemma 1. *Let L a set of n-tuples. Then L is accepted by a weakly 0-synchronized n-tape NFA if and only if $\langle L \rangle$ is regular.*

Let $\mathbb{N}$ be the set of nonnegative integers and k be a positive integer. A subset Q of $\mathbb{N}^k$ is a *linear set* if there exist vectors $v_0, v_1, \ldots, v_n$ in $\mathbb{N}^k$ such that $Q = \{v_0 + t_1 v_1 + \cdots + t_n v_n \mid t_1, \ldots, t_n \in \mathbb{N}\}$. The vectors$v_0$ (referred to as the *constant vector*) and $v_1, \ldots, v_n$ (referred to as the *periods*) are called the *generators* of the linear set Q. The set $Q \subseteq N^k$ is *semilinear* if it is a finite union of linear sets. The empty set is a trivial (semi)linear set, where the set of generators is empty. Every finite subset of $\mathbb{N}^k$ is semilinear – it is a finite union of linear sets whose generators are constant vectors. Semilinear sets are closed under (finite) union, complementation and intersection. It is known that the disjointness, containment, and equivalence problems for semilinear sets are decidable [3].

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

We will need the following result from [1]:

Theorem 1. *The emptiness (Is $L(M) = \varnothing$?) and infiniteness (Is $L(M)$ infinite?) problems for 1-tape NFA with reversal-bounded counters are decidable.*

Corollary 1. *The emptiness and infiniteness problems for n-tape NFA with reversal-bounded counters are decidable.*

An instance $I = (u_1, \ldots, u_n); (v_1, \ldots, v_n)$ of the *Post Correspondence Problem* (PCP) is a pair of n-tuples of nonnull strings over an alphabet with at least two symbols. A solution to I is a sequence of indices $i_1, i_2, \ldots, i_m$ such that $u_{i_1} \ldots u_{i_m} = v_{i_1} \ldots v_{i_m}$. It is well known that it is undecidable to determine, given a PCP instance I, whether it has a solution.

Convention: (1) We shall also refer to a set of n-tuples accepted by an n-tape machine as a language. (2) All input n-tuples $(x_1, \ldots, x_n)$ are delimited by a right end marker \$ on each tape, although sometimes the end markers are not explicitly shown. (3) A construction is *effective* if it can be implemented as an algorithm.

3 2-Ambiguous Multitape NFA

In this section, we will show that it is undecidable to determine, given a 2-ambiguous 2-tape NFA M and an integer k, whether M is weakly synchronized, whether M is weakly k-synchronized for a given k, and whether there is a weakly synchronized M' such that $L(M) = L(M')$.

We first prove this result for general 2-tape NFA and then show how to modify the proof so that it applies to the restricted case of 2-ambiguous 2-tape NFA.

Let $I = (u_1, \ldots, u_n); (v_1, \ldots, v_n)$ be an instance of the PCP. Let c and d be new symbols. We construct a 2-tape NFA M to accept the language

$$L = \{(xc^i, yd^j) \mid i, j > 0, x \neq y\} \cup \{(xc^i, xd^j) \mid i, j > 0, j = 2i, x \text{ is a solution of the PCP instance } I\}$$

as follows: M on input (xc^i, yd^j) nondeterministically selects to check (a) or (b) below:

(a) M first checks that $x \neq y$ by moving both heads in sync (0-synchronized) until it finds the first position where x differs from y (note that this also takes care of the case when their lengths are different). When M finds a discrepancy, both heads are moved to the right in sync until one head reaches the end marker and then the other head is moved to the right until it reaches the end marker. Then M accepts. Note that the whole process is deterministic.

(b) M guesses a sequence of indices $i_1, i_2, \ldots$. After guessing an index i_j, M verifies that u_{i_j} is on tape 1, and then v_{i_j} is on tape 2. This process is repeated until both x and y are exhausted. If there is no discrepancy and the heads reach the first c and d on their tapes, the second head moves right twice for every right move of the first head to check that $j = 2i$, and then M accepts.

There are six possible outcomes from a computation of M:

1. (xc^i, yd^j), where $x \neq y$ and (a) is selected: there is a 0-synchronized accepting computation of M on this input.
2. (xc^i, yd^j), where $x \neq y$, (b) is selected, and either the selected indices yield a discrepancy or $j \neq 2i$: this input will be rejected in (b).

3. (xc^i, yd^j), where $x \neq y$, (b) is selected, and the selected indices do not yield a discrepancy and $j = 2i$: this input will be accepted in (b). Note that this input will also be accepted in (a).
4. (xc^i, xd^j), and (a) is selected: this input will be rejected.
5. (xc^i, xd^j), (b) is selected, and the selected indices yield a discrepancy or $j \neq 2i$: this input will be rejected.
6. (xc^i, xd^j), (b) is selected, and the selected indices do not yield a discrepancy and $j = 2i$: this input will be accepted.

We observe the following:

Note 1: (i) From (1) and (3), it is possible that the same input of the form (xc^i, yd^j), where $x \neq y$ can be accepted in both processes (a) or (b). (ii) Another source of ambiguity is in process (b) itself – different sequences of (guessed) indices $i_1, i_2, ...$ may yield no discrepancies when matching x and y, so the same input may be accepted in many ways in (b).

Note 2: An input of the form (xc^i, xd^j), where the selected indices did not yield a discrepancy (i.e., the PCP has a solution) and $j = 2i$, is only accepted in (b) (it is rejected in (a)). Furthermore, since i is arbitrary, the heads will be out of sync unboundedly.

Hence, if the PCP instance I does not have a solution, then (xc^i, yd^j) is accepted iff $x \neq y$, so there is a 0-synchronized accepting computation (type (a)). In other words, M is weakly 0-synchronized and hence $\langle L(M) \rangle$ is regular by Lemma 1.

On the other hand, if PCP instance I has a solution, then $L(M)$ contains tuples of the form (xc^i, xd^{2i}). We claim that $L(M)$ cannot be accepted by *any* weakly
0-synchronized 2-tape NFA. If it is, then by Lemma 1, $\langle L(M) \rangle$ is regular. But then for large enough i, we can pump the string $\langle xc^i, xd^{2i} \rangle$ to get a string $\langle xc^{i+k}, xd^{i+k}d^i \rangle$ for some $k > 0$ to be in $\langle L(M) \rangle$. But $(xc^{i+k}, xd^{i+k}d^i)$ is not in $L(M)$, a contradiction.

To summarize, PCP instance I has a solution iff there is *no* weakly 0-synchronized 2-tape NFA M' such that $L(M) = L(M')$. Furthermore, the construction above shows that either M is weakly 0-synchronized or it is not weakly k-synchronized for any k. Together these results yield

Theorem 2. *The following problems are undecidable, given a 2-tape (and hence multitape) NFA M:*

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

We now modify the construction of the 2-tape NFA M above to make it 2-ambiguous. The sources of ambiguity are cited in Notes 1 and 2 above. Clearly,

since process (a) is deterministic, if we can make process (b) deterministic, then the 2-NFA will be 2-ambiguous.

We accomplish this as follows. Instead of x, we use x' where x' has two tracks: track 1 contains x and track 2 contains the "encoding" of the indices that are used to match x and y; y remains single-track. Specifically, let $I = (u_1, \ldots, u_n); (v_1, \ldots, v_n)$ be an instance of the PCP. Let $\#, e_1, \ldots, e_n$ be new symbols. For $1 \leq i \leq n$, let the string $E(i) = e_i \#^{|u_i|-1}$. Thus, the length of $E(i)$ is equal to the length of u_i. Let $\Delta = \{\#, e_1, \ldots, e_n\}$, and define the language:

$L = \{(x'c^i, yd^j) \mid i, j > 0, x'$ is a 2-track tape where the first track contains x and the second track is a string in Δ^*, $x \neq y\} \cup \{x'c^i, yd^j) \mid i, j > 0, x'$ is a 2-track tape where the first track contains x and the second track is a string $E(i_1) \cdots E(i_r)$ for some $i_1, \ldots, i_r$, $x = y$, $x = u_{i_1} \cdots u_{i_r}, y = v_{i_1} \cdots v_{i_r}, j = 2i\}$.

One can easily check that processes (a) and (b) described earlier can be made deterministic. However, it is possible that the same input of the form $(x'c^i, yd^j)$, where $x \neq y$ can be accepted in both processes (a) or (b). (x is the first track of x'.) Hence, M is 2-ambiguous. Therefore we have:

Theorem 3. *The following problems are undecidable, given a 2-ambiguous 2-tape (and hence multitape) NFA M:*

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

4 Unambiguous Multitape NFA

In this section, we show that given an unambiguous multitape NFA M and an integer k, it is decidable to determine whether M is weakly synchronized, whether M is weakly k-synchronized for a given k, and whether $L(M) = L(M')$ for some weak synchronized multitape unambiguous NFA M'. Recall that unambiguous multitape NFA have at most one accepting computation for every input n-tuple. Note that multitape DFAs are a special case. The results in this section are modifications of the corresponding results for synchronized multitape automata in [2], and we omit their proofs.

Theorem 4. *It is decidable to determine, given an unambiguous n-tape NFA M, whether it is weakly k-synchronized for some k.*

Corollary 2. *It is decidable to determine, given an unambiguous n-tape NFA M and an integer $k \geq 0$, whether M is k-synchronized.*

The following follows from the previous two results.

Corollary 3. *It is decidable to determine, given an unambiguous n-tape NFA M, whether it is weakly k-synchronized for some k. Moreover, if it is, we can effectively determine the smallest such k.*

The above results generalize to machines with reversal-bounded counters:

Theorem 5. *It is decidable to determine, given an unambiguous n-tape NFA M augmented with reversal-bounded counters, whether it is weakly k-synchronized for some k. Moreover, if it is, we can effectively determine the smallest such k.*

5 Multitape NFA on ABO-Bounded Inputs

A language is *bounded* if it is a subset of $a_1^* \cdots a_n^*$ for some distinct letters (symbols) $a_1, \ldots, a_n$. A multitape NFA is *unary* if each tape contains a string over a single symbol (letter); *bounded* if each tape contains a string from a bounded language; and *all-but-one-bounded (ABO-bounded)* if all but the first tape contains a string from a bounded language. We also refer to the inputs of such machines as unary, bounded, ABO-bounded, respectively.

In section 3, we showed that it is undecidable to determine, given a 2-tape NFA M and an integer $k \geq 0$, whether M is weakly k-synchronized. Here we show that the problem is decidable for n-tape NFA when the inputs are ABO-bounded. In fact, the result holds for n-tape NFA over $\Sigma^* \times x_{21}^* \cdots x_{2m_2}^* \times \cdots \times x_{n1}^* \cdots x_{nm_n}^*$ for some (not necessarily distinct) nonnull strings x_{ij}.

Note that if L is a set of n-tuples of strings, then $\overline{L}$ (the complement of L) is the set of n-tuples $(x_1, \ldots, x_n)$ such that $(x_1, \ldots, x_n)$ is in $\overline{L}$ if and only if $(x_1, \ldots, x_n)$ is not in L.

An n-tape NFA is *strictly k-synchronized* if in *any accepting* computation, any pair of the heads are within k cells apart (when neither head is on \$). In comparison, this condition must hold for only some accepting computation in weakly k-synchronized machines and for any computation in strongly k-synchronized machines.

Lemma 2. *Let M be an n-tape NFA that is strictly k-synchronized that accepts the language (set of n-tuples) $L = L(M)$. Then:*

1. *We can effectively construct a strictly 0-synchronized n-tape DFA M_1 accepting L.*
2. *We can effectively construct a strictly 0-synchronized n-tape DFA M_2 accepting $\overline{L}$.*

Moreover, (1) holds for n-tape weakly k-synchronized NFA as well.

Proof. For the first part, given M, we construct an ordinary (i.e., 1-tape) NFA A_1 such that $(x_1, \ldots, x_n)$ is accepted by M if and only if (the aligned version) $\langle x_1, \ldots, x_n \rangle$ is accepted by A_1. This is possible as A_1 need only maintain a finite buffer of symbols in its state. A_1 can then be converted to be deterministic (by the usual subset construction), i.e., we can convert A_1 to an equivalent DFA A_2.

Then from A_2, we can trivially construct a strictly 0-synchronized n-tape DFA M_1 accepting L.

For part 2, we can easily construct from the DFA A_2 a DFA A_3 accepting $\langle x_1, \ldots, x_n \rangle$ if and only if A_2 does not accept $\langle x_1, \ldots, x_n \rangle$. Let $L' = \{\langle x_1, \ldots, x_n \rangle \mid x_1, \ldots, x_n$ are strings with no λ's $\}$. Clearly, L' is regular and is accepted by some DFA A_4. Construct a DFA A_5 accepting $L(A_3) \cap L(A_4)$. (The reason for the intersection is to make sure that we only retain the well-formed aligned strings.) From A_5, we can then construct a strictly 0-synchronized 2-tape DFA M_2 accepting $\overline{L}$. □

We are now ready to prove the main result of this section. To illustrate the construction, we consider 3-tape NFA.

Theorem 6. *It is decidable to determine, given a 3-tape NFA M that accepts a subset of $\Sigma^* \times a^* \times b^*$ (where Σ is any alphabet and a, b are symbols) and a nonnegative integer k, whether M is weakly k-synchronized.*

Proof. Construct from M a 3-tape NFA M_1 over $\Sigma^* \times a^* \times b^*$ that is strictly k-synchronized: on input (x_1, x_2, x_3), M_1 simulates the computation of M faithfully and accepts (x_1, x_2, x_3) if M accepts (x_1, x_2, x_3) and during the computation, the heads are always within k cells apart (provided no head has reached \$).

From Lemma 2, part 2, we can effectively construct from M_1 a strictly 0-synchronized 3-tape DFA M_2 accepting $\overline{L(M_1)}$.

Clearly, M is weakly k-synchronized if and only if $L(M) \subseteq L(M_1)$ (in fact $L(M) = L(M_1)$), hence, if and only if $L(M) \cap \overline{L(M_1)} = \varnothing$, i.e., $L(M) \cap L(M_2) = \varnothing$.

To decide the above, we construct from M and M_2, a 3-tape NFA M' with four 1-reversal counters C_1, C_2, D_1, D_2 that works as follows when given input (x, a^r, b^s):

M' reads a^r and b^s and stores r in counters C_1 and D_1 and s in counters C_2 and D_2. Then M' simulates M on (x, a^r, b^s) by using counters C_1 and C_2, i,e., it decreases C_1 (resp., C_2) by 1 every time the second head (resp., third head) of M moves right on a^r (resp., b^s). At the same time, M' also simulates M_2 on (x, a^r, a^s) using counters D_1 and D_2. Note that since M_2 is 0-synchronized, D_1 (resp., D_2) is decreased only when M moves its first head to the right on x. M'accepts if M and M_2 accept.

It follows that M is weakly k-synchronized if and only if $L(M')$ is empty, which is decidable by Corollary 1. □

Corollary 4. *It is decidable to determine, given a 3-tape NFA M over $\Sigma^* \times a_1^* \cdots a_r^* \times b_1^* \cdots b_s^*$ for distinct symbols $a_1, \ldots, a_r$, $b_1, \ldots, b_s$ and a nonnegative integer k, whether M is weakly k-synchronized.*

In fact, we can prove a stronger result:

Corollary 5. *It is decidable to determine, given a 3-tape NFA M over $\Sigma^* \times v_1^* \cdots v_r^* \times w_1^* \cdots w_s^*$ for (not necessarily distinct) nonnull strings $v_1, \cdots, v_r$, $w_1, \ldots, w_s$ and a nonnegative integer k, whether M is weakly k-synchronized.*

The above corollary generalizes to multitape NFA:

Corollary 6. *It is decidable to determine, given an n-tape NFA M over $\Sigma^* \times x_{21}^* \cdots x_{2m_2}^* \times \cdots \times x_{n1}^* \cdots x_{nm_n}^*$ for some (not necessarily distinct) nonnull strings x_{ij}'s and a nonnegative integer k, whether M is weakly k-synchronized.*

6 Multitape NFA on Unary Inputs

In this section, we look at decision problems for multitape NFA on unary inputs.

6.1 Synchronizability

In Theorem 3, we saw that if the inputs of the 2-tape NFA M is unrestricted, it is undecidable to determine if there exists a weakly 0-synchronized 2-tape NFA M' equivalent to M. But what about the case when one tape has bounded input, or when both tapes have bounded inputs? At present, we do not know the answer. However, as the following shows, even for the unary case, there are machines that cannot be converted to be weakly 0-synchronized:

Proposition 1. *$L = \{(a^m, b^n) \mid m > 0, n = 2m\}$ can be accepted by a 2-tape NFA M but cannot be accepted by a weakly 0-synchronized 2-tape NFA.*

6.2 Weakly Synchronized Regular Languages

First we consider the extension of the definition of strongly k-synchronized and weakly k-synchronized to *languages* (instead of *machines* in the original definitions) over a binary alphabet $\Sigma = \{a, b\}$ and show that whether or not a regular language is weakly synchronized is decidable. We do so via a structural characterization of weakly synchronized regular languages, which is of independent interest.

A word w is *strongly k-synchronized* if for any factorization $x = uv$

$$-k \leq |u|_a - |u|_b \leq k. \tag{1}$$

A language L over Σ is *strongly k-synchronized* if all of its words are strongly k-synchronized, and *strongly synchronized* if it is strongly k-synchronized for some k. L is called *weakly k-synchronized* if for every $w \in L$, there is a corresponding $w' \in L$ such that $P(w') = P(w)$ and w' is strongly k-synchronized. L is *weakly synchronized* if it is weakly k-synchronized for some k. Suppose M is a DFA over $\Sigma = \{a, b\}$. Consider an r-cycle C in M given by the sequence of states

$$q_1, q_2, \ldots, q_{r+1} \tag{2}$$

with $r \geq 2$ and $q_1 = q_{r+1}$. The word associated to C is $a_1 a_2 \cdots a_r$ where $\delta(q_i, a_i) = q_{i+1} \quad (i = 1, 2, \ldots, r)$. When there is no ambiguity, we use C to also denote the associated word $a_1 a_2 \ldots a_r$. We call C *balanced* if $|C|_a = |C|_b$. C is *a-heavy* if $|C|_a > |C|_b$, and *b-heavy* if $|C|_a < |C|_b$. C is a *simple* cycle if $q_i \neq q_j$ $(i, j = 1, 2, \ldots, r)$ in (2).

Lemma 3. *Suppose M is a DFA with m states with no unbalanced simple cycles. If C is a cycle in M then the word C is strongly m-synchronized.*

Theorem 7. *L is weakly synchronized iff the minimum state DFA M for L has no unbalanced simple cycles.*

Proof. Suppose M has an unbalanced simple r-cycle C. We will show that L is not weakly synchronized. WLOG C is a-heavy. By the minimality of M, the beginning state q_1 of C is reachable from the initial state of M, and there is a path from $q_{r+1} = q_1$ to a final state of M. It follows that we can pump C: i.e. there exists words x, y such that for every $t \geq 0$, $w_t = xC^t y \in L$. Then $P(w_t) = (c_0 + c_1 t, d_0 + d_1 t)$ for constants $c_0, c_1, d_0, d_1 \geq 0$ with $c_1 > d_1$. Therefore for any word w'_t with $P(w_t) = P(w'_t)$, taking $u = w'_k$, $|u|_a - |u|_b = c_0 - d_0 + (c_1 - d_1)t \to \infty$ as $t \to \infty$, so (1) cannot hold for any fixed k. Therefore L is not weakly synchronized. Conversely, assume that every simple cycle in M is balanced. We will show that L is strongly synchronized, and therefore weakly synchronized. Let m be the number of states of M. We show that L is strongly $2m$-synchronized. Any w accepted by M can be written as $w = x_0 C_1 x_1 C_2 \dots C_t x_t$ where each C_i is a balanced cycle (not necessarily simple) and $|x_0 x_1 \cdots x_t| < m$. By lemma 3, each cycle is strongly m-synchronized. Since the contribution of the part $x_0 x_1 \cdots x_t$ to the difference of the number of occurrences of a's and b's in w is at most m, any prefix u of w satisfies (1) with $k = 2m$. □

Corollary 7. *A regular language L is strongly synchronized iff it is weakly synchronized.*

Corollary 8. *It is decidable whether a regular language L is weakly synchronized.*

Finally, we can state the condition for the weak synchronizability of a regular language L in terms of its Parikh image. Consider a linear set that appears in $P(L)$: $\{(a_0, b_0) + k_1(a_1, b_1) + \cdots + k_s(a_s, b_s) \mid k_1, k_2, \dots, k_s \geq 0\}$. Each of the vectors $(a_i, b_i), i = 1, \dots, s$ are called *periods*. A period (a_i, b_i) is balanced iff $a_i = b_i$. If $a_i > b_i$, then the period is a-heavy, if $b_i > a_i$ then it is b-heavy. These notions are translations of the ones on cycles to the Parikh images where a balanced period corresponds to a balanced simple cycle, etc. Therefore

Corollary 9. *For a regular language L over $\Sigma = \{a, b\}$, the following are equivalent:*

1. *L is strongly synchronized,*
2. *L is weakly synchronized,*
3. *The minimum state DFA for L has no unbalanced simple cycles,*
4. *The Parikh image $P(L)$ has no unbalanced periods.*

6.3 Weakly Synchronized NFA on Unary Inputs

In Section 3, we showed that it is undecidable to determine, given a 2-tape NFA M, whether it is weakly k-synchronized for some k. The problem is also

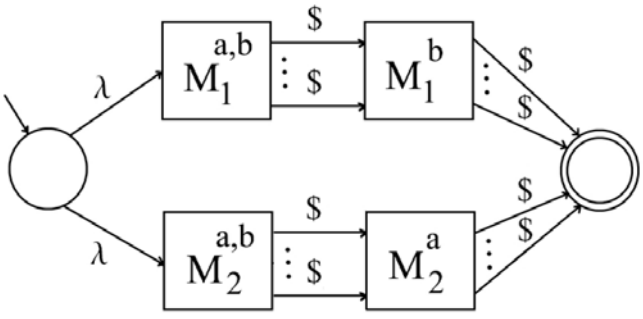

Fig. 1. An ordinary NFA that accepts accepting computations of a 2-tape NFA

undecidable when k is specified, even for $k = 0$. In Section 5, we showed that it is decidable to determine, given a 2-tape NFA M one of whose tapes contains a string over a bounded language and an integer k, whether it is weakly k-synchronized. We currently do not know if this restriction would make determining whether a 2-tape NFA M is weakly k-synchronized for some unspecified k decidable. It appears to be a difficult combinatorial problem. However, for the special case when the two tapes are unary, we can show that the problem is decidable.

Let M be a 2-tape NFA where the inputs are of the form $(a^n\$, b^m\$)$. We assume that exactly one of the heads moves during the computation of M.

The transitions of M can be labeled with letters from $\Sigma = \{a, b, \$\}$. A head movement on the first (second) tape is labeled with a (b). The last move by each head is labeled \$. In this way each computation path of M can be identified with a word over Σ. Each such word contains exactly two occurrences of the symbol \$. We will not consider these when we look at the Parikh vectors of the words accepted by M. Since M is nondeterministic, there may be many accepting computation paths w for an accepted input, but each of these has Parikh vector $P(w) = (a^n, b^m)$.

M can be trivially modified into an ordinary NFA M' that accepts the accepting computations of M as shown in Figure 1.

There are two types of accepting computations:

1. $w = x\$y\$$ with $x \in \{a, b\}^*$ and $y \in \{b\}^*$,
2. $w = x\$y\$$ with $x \in \{a, b\}^*$ and $y \in \{a\}^*$.

The first type comes from accepting computations in which the first head reaches \$ first, and then the computation continues on tape two with the second head moving to the right consuming b's until it reaches \$ on the second tape. The second type is similar, with the roles of the two tapes interchanged. Consequently, whether M is weakly k-synchronized is equivalent to showing that for any given word $w \in L(M')$, there is a strongly k-synchronized word u such that either $ub^j \in L(M')$ and $P(w) = P(u)+(0, j)$, or $ua^i \in L(M')$ and $P(w) = P(u)+(i, 0)$.

Theorem 8. *Suppose M is a unary 2-tape NFA. Then it is decidable whether or not M is weakly synchronized.*

Proof. We can assume that exactly one of the heads moves one cell to the right in each step until the heads reach \$. Let $L = L(M)$. We can assume that the automata that appear in the boxes in Figure 1 are minimum state DFA. The language accepted by the upper part of Figure 1 can be written as the disjoint union of languages accepted by pairs of DFA, corresponding to final states of $M_1^{a,b}$. If $M_1^{a,b}$ has t final states, then this results in t pairs of minimum state DFA $(F_i^{a,b}, N_i^b)$. The corresponding language accepted is the concatenation $L(F_i^{a,b})L(N_i^b)$. There is a similar decomposition for the language accepted by the lower part of Figure 1. For any given i, consider $P_i = P(F_i^{a,b})$. Eliminating each unbalanced period from each of the linear sets in P_i results in a new Parikh image P_i'. This is the Parikh image of the automaton obtained from $F_i^{a,b}$ by eliminating unbalanced cycles as follows.

Consider a simple unbalanced r-cycle C of $F_i^{a,b}$ given by $q_1, q_2, \ldots, q_{r+1}$ with $r \geq 2$ and $q_1 = q_{r+1}$. Let $a_1 \cdots a_r \in \Sigma^*$ be the word associated where $\delta(q_i, a_i) = q_{i+1}$ $(i = 1, 2, \ldots, r)$ Consider the new alphabet $\Sigma' = \Sigma \cup \{x_1, x_2, \ldots, x_r\}$. Alter the labels on C by replacing a_i $(a_i \in \{a, b\})$ by the symbol x_i. Denote by L_{01} the language of all words over Σ' taking the start state of $F_i^{a,b}$ to q_1, and by L_{1f} the language of all words over Σ' taking $q_{r+1} = q_1$ to the final state of $F_i^{a,b}$. If h is the homomorphism defined by $h(x_i) = a_i$, $(i = 1, \ldots, r)$, then the language

$$h\left(L(F_i^{a,b}) \setminus L_{01}x_1x_2\cdots x_rL_{1f}\right)$$

is the language of words in $L(F_i^{a,b})$ which are accepted without traversing the cycle C. Since there are finitely many simple cycles in $F_i^{a,b}$, in finitely many steps we can take away from $L(F_i^{a,b})$ all of the words that traverse an unbalanced cycle of $F_i^{a,b}$. This has the effect of deleting all of the unbalanced periods from the linear sets that appear in the Parikh map.

Let P_i' be the Parikh image of the resulting automaton, and let P^u be the union of the sets $P_i' + P(N_i^b)$ over i. For the lower part of the diagram, a similar construction results in the set P^l. By the characterization of weak synchronization preceding the statement of Theorem 8, M is weakly synchronized iff $P(L) \subseteq P^u \cup P^l$. Each of the Parikh images above can effectively be constructed and the containment required can be effectively checked, since Parikh images of regular languages are semilinear. □

7 Conclusion

We looked at decision questions concerning weak synchronization of the heads of multitape NFA. Most of our results are the strongest possible in the sense that slight restrictions on the models make the undecidable problems decidable.

Some questions remain open. In particular, is it decidable to determine, given a 2-tape NFA whose tapes are over bounded languages, whether it is weakly k-synchronized for some k ? This question seems quite difficult – we have only been able to resolve this question (in the positive) for the bounded unary case.

References

1. Ibarra, O.H.: Reversal-bounded multicounter machines and their decision problems. J. Assoc. Comput. Mach. 25, 116–133 (1978)
2. Ibarra, O.H., Tran, N.: On synchronized multitape and multihead automata. In: Proc. of the 13th Int. Workshop on Descriptional Complexity of Formal Systems (DCFS 2011), pp. 184–197 (2011)
3. Ginsburg, G., Spanier, E.: Bounded Algol-like languages. Trans. of the Amer. Math. Society 113, 333–368 (1964)
4. Parikh, R.J.: On context-free languages. J. Assoc. Comput. Mach. 13, 570–581 (1966)
5. Yu, F., Bultan, T., Cova, M., Ibarra, O.H.: Symbolic String Verification: An Automata-Based Approach. In: Havelund, K., Majumdar, R. (eds.) SPIN 2008. LNCS, vol. 5156, pp. 306–324. Springer, Heidelberg (2008)
6. Yu, F., Bultan, T., Ibarra, O.H.: Symbolic String Verification: Combining String Analysis and Size Analysis. In: Kowalewski, S., Philippou, A. (eds.) TACAS 2009. LNCS, vol. 5505, pp. 322–336. Springer, Heidelberg (2009)
7. Yu, F., Bultan, T., Ibarra, O.H.: Relational String Verification Using Multi-track Automata. In: Domaratzki, M., Salomaa, K. (eds.) CIAA 2010. LNCS, vol. 6482, pp. 290–299. Springer, Heidelberg (2011)