Journal of Universal Computer Science, vol. 8, no. 2 (2002), 332-347
submitted: 15/9/01, accepted: 27/1/02, appeared: 28/2/02 © J.UCS

Generation of Constants and Synchronization of Finite Automata[1]

Arto Salomaa
(Turku Centre for Computer Science Lemminkäisenkatu 14A
20520 Turku, Finland
Email: asalomaa@utu.fi)

Abstract: The problem about the *synchronization* of a finite deterministic automaton is not yet properly understood. The present paper investigates this and related problems within the general framework of a composition theory for functions over a finite domain N with n elements. The notion of *depth* introduced in this connection is a good indication of the *complexity* of a given function, namely, the complexity with respect to the length of composition sequences in terms of functions belonging to a basic set. The depth may vary considerably with the target function. Not much is known about the reachability of some target functions, notably constants. *Synchronizability* of a finite automaton amounts to the representability of some constant as a composition of the functions defined by the input letters. Properties of n such as primality or being a power of 2 turn out to be important, independently of the semantic interpretation. We present some necessary, as well as some sufficient, conditions for synchronizability. We also discuss a famous conjecture about the length of the shortest synchronizing word, and present some results about *universal* synchronizing words.

Key Words: synchronization of finite automata, functions over a finite domain, functional compositions, complexity of compositions

Category: F.1.1

1 Functions over a finite domain

In this paper we will consider functions $g(x)$ whose domain is a fixed finite set N with n elements, $n \geq 2$, and whose range is included in N. We will mostly deal with this abstract setup. It is clear that such a setup occurs in many and very diverse situations, interpretations. Depending on the interpretation, different questions will be asked.

The two interpretations mostly studied in the past are *many-valued logic* and *finite automata.* In the former, the set N consists of n *truth values* and the functions are truth functions. In the latter, the set N consists of the *states* of a finite automaton, whereas each letter of the input alphabet induces a specific function: the next state when reading that letter.

We will restrict the attention to functions with *one* variable only. In many contexts, especially in many-valued logic, it is natural to consider functions of several variables. Many of the considerations below have been extended to functions of several variables in [16, 17, 18].

[1] C. S. Calude, K. Salomaa, S. Yu (eds.). *Advances and Trends in Automata and Formal Languages. A Collection of Papers in Honour of the 60th Birthday of Helmut Jürgensen.*

We make the following **convention**, valid throughout this paper: n always stands for the number of elements in the basic set N. In most cases we let N simply be the set consisting of the first n natural numbers:

$$N = \{1, 2, \ldots, n\}.$$

Clearly, there are altogether n^n functions in the set N^N we are considering.

Consider a couple of examples. If we are dealing with n-valued logic, and the function g is defined by the equation

$$g(x) = n - x + 1, x = 1, 2, \ldots, n,$$

then g is the well-known *Łukasiewicz negation.* (1 is the truth value "true", n is the truth value "false", whereas the other numbers represent the intermediate truth values.) If we are dealing with a finite deterministic automaton whose state set equals N, the function g defined by the equation above could be viewed as transitions affected by a specific input letter a. Under this interpretation, the letter a interchanges the states n and 1, the states $n-1$ and 2, and so forth. When we speak of "functions", without further specifications, we always mean functions in the setup defined above. Clearly, the *composition ab* of two functions a and b is again a function. We read compositions from *left to right:* first a, then b. This is in accordance of reading the input words of a finite deterministic automaton from left to right.

Our point of departure will be a nonempty set $\mathbf{F}$ of functions. The only assumption about the set $\mathbf{F}$ is that it is a nonempty subset of the set N^N of all functions; $\mathbf{F}$ may consist of one function or of all functions. We will consider the set $\mathbf{G(F)}$ of all functions *generated* by $\mathbf{F}$, that is, obtained as compositions (with arbitrarily many composition factors) of functions from $\mathbf{F}$. If a particular function f can be expressed as a composition of functions $a_i, i = 1, 2, \ldots, k$, belonging to $\mathbf{F}$:

$$f = a_1 a_2 \ldots a_k,$$

where some of the functions a_i may coincide, then the word $a_1 a_2 \ldots a_k$ is referred to as a *composition sequence* for f. The number k is referred to as the *length* of the composition sequence. The function f is often referred to as the *target function.* Observe that our composition sequences have to be nonempty, implying that the identity function is not necessarily in $\mathbf{G(F)}$; it is in there exactly in case the set $\mathbf{F}$ contains at least one permutation.

Clearly, $\mathbf{G(F)}$ can be viewed as the *semigroup* generated by $\mathbf{F}$. However, we will prefer the more straightforward approach and will not use semigroup-theoretic terminology in the sequel.

The set $\mathbf{F}$ is termed *complete* if all of the n^n functions are in $\mathbf{G(F)}$. Following [12], we will speak also of the *genus* and *type* of a function f. A function f is said to be of *genus t* if it assumes exactly t values. Thus, the genus equals the cardinality of the range of f. A function f of genus t is said to be of *type* $m_1 \oplus m_2 \oplus \ldots \oplus m_t$, where $m_1 + m_2 + \ldots + m_t = n$ if, for each i with $1 \leq i \leq t$, there is a number y_i such that f assumes y_i as a value exactly m_i times. Obviously we do not change the type if we change the order of the numbers m_i, which means that the operation $\oplus$ is commutative. For instance, permutations are of genus n and of type $1 \oplus 1 \oplus \ldots \oplus 1$. The type of a function f tells us how many values f assumes and how many times it assumes each value. It does not tell us what these values are and in what order they are assumed.

Since n is finite, a specific function f can always be defined by a table. Omitting the argument values, this amounts to giving the *value sequence* of f, that is, the sequence $f(1),\ f(2),\ldots,\ f(n)$ of its values for the increasing values of the argument. The Łukasiewicz negation can be defined in this way by its value sequence

$$n,\ n-1,\ldots,2,\ 1.$$

When there is no danger of confusion, we omit the commas from the value sequence. Thus, for $n = 6$, the value sequence of the Łukasiewicz negation reads 654321.

Completeness (of a set of functions) is fairly well understood, whereas rather little is known about the *length* of the shortest composition sequence for a given target function f. As will be indicated in more detail below, the length can be viewed as the *complexity* or *depth* of f. For instance, [10] gives a comprehensive account about the bases of the symmetric group. However, given a basis B and a permutation p generated by B, very little is known about the length of the product of the elements of B needed to generate p. General problems of this nature are NP-hard.

To give a flavor of the completeness results, we now list some facts without proofs. Further discussion can be found in [13, 14, 15, 17, 18]. We use the simple expression "a set **F** of functions *generates* another set **F'** of functions" to mean that **F'** is contained in $\mathbf{G(F)}$.

Lemma 1 *Assume that* $n \geq 4$ *and* f *is of genus* $< n$. *Then the set consisting of* f *and of the members of the alternating group* A_n *generates every function of the same type as* f.

Lemma 2 *The set of all functions of type* $m_1 \oplus m_2 \oplus \ldots \oplus m_t$, *where* $1 < t < n$, *generates every function of type* $(m_1 + m_2) \oplus \ldots \oplus m_t$.

The proofs of the two lemmas are not difficult. A minor difficulty in the former is to use only *even* permutations in the construction. The lemmas indicate the possibilities for constructing new functions if adequate permutations are available. For instance, by Lemma 2, all constants are generated by the set given in Lemma 1.

It is often possible to show that a composition sequence cannot any more be continued to yield a given target function. Thus, in many cases, a wrong choice at the beginning destroys the possibilities of obtaining the target function. For instance, if the numbers in the type of a composition sequence are all even, then the sequence cannot be continued to yield a target function whose type contains an odd number. Similarly, a composition sequence of type $3 \oplus 2$ cannot be continued to yield a target function of type $4 \oplus 1$. On the other hand, if the target function is a *constant*, then the situation is entirely different: wrong choices may be corrected later on. In other words, if a composition sequence for a constant is at all possible, a composition sequence can also be obtained by continuing an arbitrary prefix. Constants are the only functions having this property. This is obvious also in view of the following lemma, which contains some simple observations along these lines.

Lemma 3 *The genus of the composition* fg *exceeds neither the genus of* f *nor the genus of* g. *Assume that* f *is of type* $m_1 \oplus \ldots \oplus m_t$. *Then the type of the* *composition fg is obtained by addition: each number in the latter type is the sum of some of the numbers m_i, whereby each of the numbers m_i is used exactly once.*

For the proof of the following main *completeness theorem for unary functions*, we refer to [15].

Theorem 1 *Given a nonidentical function f of genus n and a function g of genus $n-1$, a function h can be effectively constructed such that the set $\{f, g, h\}$ is complete, provided it is not the case that $n = 4$ and f is one of the three permutations (12)(34), (13)(24) and (14)(23). For $n \geq 3$, no less than three functions suffice to generate all functions and, whenever three functions form a complete set, then two of them constitute a basis of the symmetric group S_n and the third is of genus $n-1$.*

An essential tool in the proof is a result by Piccard (see [10], pp.80-86), according to which a companion can be found to any nonidentical permutation such that the two permutations constitute a basis of the symmetric group S_n. The exceptional case here is $n = 4$: no permutation in the Klein Four-Group can be extended to a basis of S_4.

We conclude this section by discussing explicitly how the synchronizability of finite automata fits into our general framework. The classical paper by E.F. Moore, [9], about *Gedanken experiments* on finite automata, had the general idea to view a finite automaton as a black box and to try to find out some specific facts about it by observing what kind of outputs certain inputs produced. Of course, for each experiment, the overall setup has to be defined explicitly. (See [6] for an early contribution.) Suppose you know the structure (graph, transition function) of a given finite deterministic automaton A, but do not know the state A is in. How can you get the situation under control? For some automata, not always, there are words, referred to as *synchronizing*, bringing the automaton always to the same state q, no matter from which state you started from. Thus, you first have to feed A a synchronizing word, after which you have the situation completely under control. You can also view the graph of an automaton as a labyrinth, where you are lost. If you then follow the letters of a synchronizing word (and have the global knowledge of the graph of the automaton), you have found your way. This shows the connection with the well-known *road coloring* problem, [1].

Clearly, a synchronizing word can be viewed as a *composition sequence for a constant*, and we are back in the setup introduced above. Indeed, consider a finite deterministic automaton, without initial and final states, as a pair $(N, \mathbf{F})$, where N is the state set of cardinality n and $\mathbf{F}$ is a set of functions mapping N into N. The set $\mathbf{F}$ determines both the input alphabet and the transition function in the natural way, and input words correspond to compositions of functions. Our convention about reading compositions from left to right is in accordance with the customary way of reading input words from left to right.

An automaton is *synchronizable* if and only if it possesses a synchronizing word. This happens exactly in case a constant function is in $\mathbf{G(F)}$.

2 Depth of compositions

We now come to the central notions concerning the length of composition sequences. For any language L, we denote by $\min(L)$ the length of the shortest word in L. (If L is empty, we agree that $\min(L) = \infty$.) We denote by $L(\mathbf{F}, f)$ the set of all composition sequences for f, that is, the *language* over the alphabet $\mathbf{F}$ whose words, viewed as composition sequences, yield the function f.

Definition 1 *The* depth *of a function f* with respect *to the set $\mathbf{F}$, in symbols $D(\mathbf{F}, f)$, is defined by the equation*

$$D(\mathbf{F}, f) = \min(L(\mathbf{F}, f)).$$

Thus, the depth of a function with respect to a particular set can also be ∞.

The depth of a function f *is defined by the equation*

$$D(f) = \max(D(\mathbf{F}, f)),$$

where $\mathbf{F}$ ranges over all sets with the property $L(\mathbf{F}, f) \neq \emptyset$.

Because, for any f, there are sets $\mathbf{F}$ with the property mentioned in the definition, we conclude that the depth of a function is always a positive integer. (The notion of depth was introduced in [8], where it was referred to as "complexity".)

Definition 2 *The* complete depth $D_C(f)$ *of a function f is defined by the equation*

$$D_C(f) = \max(D(\mathbf{F}, f))$$

where now $\mathbf{F}$ ranges over complete *sets of functions.*

In the latter definition it is *a priori* clear that $L(\mathbf{F}, f) \neq \emptyset$.

It follows by the definitions that every function f satisfies

$$D_C(f) \leq D(f).$$

However, lower bounds are much harder to obtain for $D_C(f)$, for the simple reason that we have much less leeway if we have to restrict the attention to complete sets $\mathbf{F}$ only.

In the sequel, we are especially interested in the *constants*

$$c_i(x) = i, \text{ for all } x \text{ and } i = 1, 2, \ldots, n.$$

We use the notation SYNCHRO for the class of all sets $\mathbf{F}$ such that at least one of the constants c_i is in $\mathbf{G(F)}$. (In defining SYNCHRO we have some fixed n in mind. Thus, SYNCHRO actually depends on n.) Analogously to the definitions above, we now define the depths

$$D(\text{const}),\ D_C(\text{const}).$$

By definition,

$$D(\text{const}) = \max_{\mathbf{F}} \min\{D(\mathbf{F}, c_i) | 1 \leq i \leq n\},$$

where $\mathbf{F}$ ranges over SYNCHRO. The depth $D_C(\text{const})$ is defined in exactly the same way, except that $\mathbf{F}$ ranges over complete sets. Thus, the notions introduced do not deal with the depth of an individual function but rather give the smallest depth of a constant, among the constants generated. Similarly as SYNCHRO, the notions depend on n.

We now show that there are specific functions and classes of functions whose depth cannot be bounded from above by a polynomial in n. We give the basic definition in automata-theoretic terms. Coming back to Gedanken experiments, [9], we might sometimes want to keep the black box as it was, unchanged. This leads to the following definition. A *stabilizing* word might give some information about the behavior of the automaton, especially if the latter has outputs, but in any case we do not lose anything since we always return to the starting point.

Definition 3 *A nonempty word w over the alphabet Σ of a finite deterministic automaton (Q, Σ, δ) is* stabilizing *if $\delta(q, w) = q$ holds for all states q. Similarly, w is* transposing *if there are two distinct states q_1 and q_2 such that*

$$\delta(q_1, w) = q_2,\ \delta(q_2, w) = q_1,\ \delta(q, w) = q \text{ for } q \neq q_1, q_2.$$

Observe that, from the point of view of functions over the finite domain $N = Q$, a stabilizing word defines the *identity* function, whereas transposing words correspond to a *class* of functions. Analogously to the notation SYNCHRO, we use the notations STABIL and TRANSP. Thus, TRANSP stands for the class of all sets $\mathbf{F}$ such that at least one of the transposing functions is in $\mathbf{G(F)}$. The depths

$$D(\text{stabil}),\ D_C(\text{stabil}),\ D(\text{transp}),\ D_C(\text{transp})$$

are defined in the same way as for SYNCHRO and const. For instance,

$$D(\text{transp}) = \max_{\mathbf{F}} \min\{D(\mathbf{F}, t) | t \text{ is transposing}\},$$

where $\mathbf{F}$ ranges over TRANSP. All of these notions depend on n.

Theorem 2 *There is no polynomial $P(n)$ such that $D(\text{stabil}(n)) \leq P(n)$, for all n. There is no polynomial P(n) such that $D(\text{transp}(n)) \leq P(n)$, for all n.*

Proof. It suffices to consider an infinite sequence of numbers n of a specific form. Consider first the case of STABIL. Let p_i be the ith prime, and consider numbers n of the form

$$n = p_1 + p_2 + \cdots + p_k.$$

Let a be a permutation in the symmetric group S_n, defined as the product of k cycles of lengths $p_1, p_2, \ldots, p_k$. Let the set $\mathbf{F}$ consist of a only. Let the target function id be the identity function. Clearly,

$$D(\mathbf{F}, id) = p_1 p_2 \cdots p_k = \Pi,$$

where the last equality is only a notation. By the well-known estimate $p_k \leq k^2$, we obtain $n \leq kp_k \leq k^3$, whence $k \geq \sqrt[3]{n}$. Since obviously $\Pi \geq k!$, we obtain finally

$$D(id) \geq [\sqrt[3]{n}]!$$

which shows that $D(\text{stabil}(n))$ cannot have any upper bound $P(n)$. (Observe that the situation described above can be viewed as an automaton with only one input letter a. A similar technique has often been used in connection with trade-offs between deterministic and nondeterministic finite automata. The automaton is disconnected: the states form cycles with prime lengths. However, the automaton can easily be replaced by a connected one by adding a dummy letter not affecting the shortest stabilizing word.)

The case of TRANSP is handled in almost the same way. We just omit the first prime $p_1 = 2$ from the product Π, obtaining the product Π'. Since Π' is odd, the word $w' = a^{\Pi'}$ interchanges the two states in the first cycle. For this particular automaton, the word w' is the shortest transposing word. The above argument is now applicable for upper bounds of $D(\text{transp}(n))$, because division by 2 does not affect any of the conclusions.

Corresponding questions about the complete depth, such as whether or not $D_C(\text{transp}(n))$ possesses a polynomial upper bound, remain open. If one has to deal with functionally complete automata, lower bounds such as the ones in the above proof are hard to obtain.

3 Basic problems about constants

We discussed in Section 1 the interconnection between the synchronizability of a finite automaton and a constant belonging to $\mathbf{G(F)}$. Although very basic and studied from the early days of automata theory, no satisfactory solution is known for the following problems.

- Characterize synchronizable automata. (Find a criterion telling when a constant is in $\mathbf{G(F)}$.)
- What is the length of the shortest synchronizing word? (Given a set $\mathbf{F}$ in SYNCHRO, what is the length of the shortest composition sequence for a constant?)

We discuss in this section the latter problem. As regards the former problem, we will present in Sections 4 and 5 some sufficient, as well as some necessary conditions for synchronizability. *Universal synchronizing words* will be investigated in the final Section 6.

The well-known Černý Conjecture can be expressed in the following form.

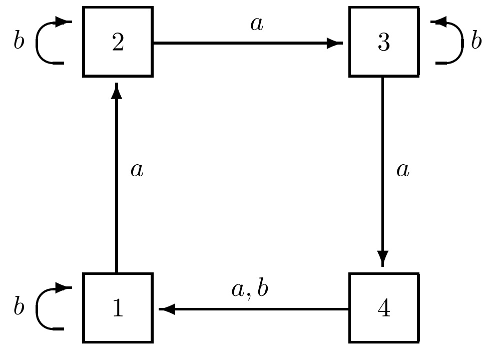
Conjecture 1 (Černý) $D(\text{const}) = (n-1)^2$.

We now present some facts related to Conjecture 1. The first question is: Could one do better, that is, obtain a still smaller depth? The answer is negative, as shown by the following example.

Consider the finite deterministic automaton $\mathcal{A}'$ defined as follows. The automaton $\mathcal{A}'$ has the state set N, and two input letters a (affects a circular permutation) and b (identity except sends n to 1). The transitions are defined by the table

$$\begin{array}{c|cccc} \delta & 1 & 2 \ldots & n-1 & n \\ \hline a & 2 & 3 \ldots & n & 1 \\ b & 1 & 2 \ldots & n-1 & 1 \end{array}$$

We still depict the automaton for $n = 4$.



The word $(ba^{n-1})^{n-2}b$ is synchronizing and, moreover, there are no shorter synchronizing words and this word is the only synchronizing word of length $(n-1)^2$. The reader might want to prove these facts. The first impression is that there could be shorter synchronizing words. Indeed, there are many possibilities of applying a "greedy" algorithm, that is, using the "reduction" b earlier than after $n-1$ uses of the circular a. (Clearly, b has to be used in the reduction sense altogether $n-1$ times.) But if one does so, one has to pay a price later on, which makes the indicated synchronizing word shortest. The automaton $\mathcal{A}'$ was presented in [4]. For further information, see [3, 8, 11, 16, 17].

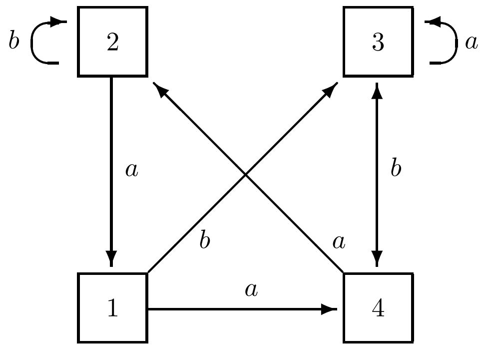
Let $\mathbf{F}$ consist of the functions a and b defining this particular automaton $\mathcal{A}'$, and consider the constant functions $c_i,\ 1 \le i \le n$.

Lemma 4 *$D(\mathbf{F}, c_i) = (n-1)^2 + i - 1$, for $i = 1, 2, \ldots, n$.*

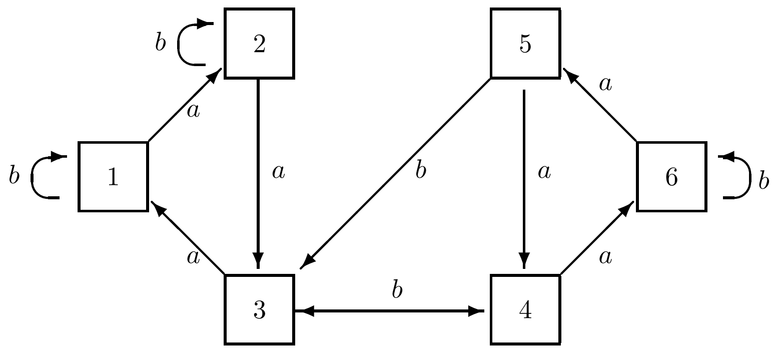
See [15] for a proof of the lemma. The lemma yields the lower bounds in the following theorem. The reference [15] contains also a proof of the upper bound; various cubic upper bounds are known in the literature for different setups. The upper bound $2^n - n - 1$ (agreeing with $(n-1)^2$ for $n = 2, 3$) follows directly by the definition: if a synchronizing word w can be decomposed as $w = xyz$, where $\delta(Q, x) = \delta(Q, xy)$, then also the word xz is synchronizing.

Theorem 3 *For a constant c_i, we have $n(n-1) \le D(c_i) < n^3/2$. Moreover, $D(\mathrm{const}) \ge (n-1)^2$.*

Thus, Conjecture 1 is actually open only as regards the upper bound. (Actually, a natural generalization of Conjecture 1 to functions of several variables is not valid, [16, 17].) On the other hand, the automaton defined above is the only automaton (when isomorphic variants are disregarded) known, which is defined for an arbitrary n and which reaches the lower bound. For $n = 3$, there are many automata reaching the lower bound 4. For $n = 4$, also the following automaton due to [5] reaches the lower bound 9:



Moreover, the following automaton due to [7], with $n = 6$, reaches the bound.



The shortest synchronizing words are of length $(6-1)^2 = 25$:

$$baababababaabbababaabbabaab, \; baababababaabbabaabaababaab.$$

Conjecture 1 is very strange in the sense that it is very difficult to find a counter example but it is also *hard to find examples satisfying the Conjecture in the strict sense*, that is, examples where the bound $(n-1)^2$ is actually reached. Indeed, the examples mentioned above are the only ones known to me for values $n \geq 4$.

The variant of Conjecture 1, dealing with the depths of individual functions, can be expressed as follows.

Conjecture 2 *If c_i is a constant, then $D(c_i) = n(n-1)$.*

It is a consequence of Theorem 3 that the depth $n(n-1)$ cannot be reduced. If Conjecure 1 holds, so does Conjecture 2. This is a consequence of the following simple result.

Lemma 5 *For a constant c_i, we have $D(c_i) \leq D(\text{const}) + (n-1)$.*

On the other hand, it is conceivable (although very unlikely) that Conjecture 2 holds but Conjecture 1 fails.

It is not known whether the upper bound ($(n-1)^2$ or $n(n-1)$) is reached for the complete depth. Thus, it is possible that

$$D_C(\text{const}) < (n-1)^2,$$

and $D_C(c_i) < n(n-1)$ holds for the constant c_i.

Conjecture 3 $D_C(\text{const}) \leq (n-1)^2 - (n-3)$. *Moreover, the complete depth of a constant c_i satisfies*

$$D_C(c_i) \leq (n-1)^2 + 2.$$

If true, Conjecture 3 gives also an example of a function whose complete depth is strictly smaller than its depth. Various claims have been made in the literature about the validity of Conjecture 1 for "small" values of n. The Conjecture clearly holds for $n = 2, 3$ because in this case the bound $(n-1)^2$ coincides with the bound obtained from the number of all subsets. Really convincing proofs are missing from the other cases. The number of possibilities is huge. For $n = 5$, one has to go through 2^{5^5} subcases.

4 Sufficient conditions for synchronizability

We have already pointed out that no characterization of synchronizable finite automata is known. Some necessary and sufficient conditions, implied almost directly by the definition, can be given. For instance, an automaton with the transition function δ is synchronizable exactly in case, for any pair (p, q) of states, there is a word x with the property $\delta(p, x) = \delta(q, x)$. However, testing synchronizability according to such a criterion is hard. In this and the next section we give some less obvious criteria that either guarantee synchronizability or non-synchronizability.

Thus, the problem is whether or not a given set $\mathbf{F}$ of functions is in SYNCHRO. Since a set consisting of permutations only can never be in SYNCHRO, we denote by PRESYNCHRO the collection of all sets $\mathbf{F}$ containing at least one function of genus less than n. Thus the problem consists of finding a characterization for those sets in PRESYNCHRO that are in SYNCHRO.

It is an immediate consequence of Lemmas 1 and 2 that, whenever $\mathbf{F}$ is in PRESYNCHRO and contains the alternating group, then $\mathbf{F}$ is in SYNCHRO. The following result is somewhat stronger.

Theorem 4 *If a set in* PRESYNCHRO *contains a doubly transitive group, then it is in* SYNCHRO.

Proof. Let $\mathbf{F}$ be the given set and g a function in $\mathbf{F}$ of genus $< n$. Hence, there are two distinct numbers i and j such that $g(i) = g(j)$. Consequently, whenever f is a function whose range contains the numbers i and j, then fg is of smaller genus than f. Suppose we have a composition sequence w for a function of genus $t > 1$. We take two arbitrary numbers i_1 and j_1 from the range of this function, as well as a permutation h from the doubly transitive group mapping the pair (i_1, j_1) onto the pair (i, j). Now the composition sequence whg defines a function of genus $< t$. Continuing in the same way, a constant (genus 1) is obtained, which concludes the proof.

Observe that the proof does not directly yield any estimates for the *length* of the composition sequence for a constant, because the doubly transitive group might be given in terms of some generators and the required permutations h might have long composition sequences.

The above result cannot be extended to concern (simply) transitive groups. Indeed, it is not necessarily the case that a set belonging to PRESYNCHRO and containing a circular permutation is in SYNCHRO. This will be seen in the next section.

Properties of n, such as primality or being a power of 2, influence the theory of compositions. (See, for instance, [14].) We now show that, if n is prime, then a set in PRESYNCHRO that contains a circular permutation actually is in SYNCHRO.

The following lemma can be viewed as folklore, see [18]. We give the proof, since it is not quite straightforward.

Lemma 6 *Assume f is a circular permutation and $N = N_1 \cup \ldots \cup N_t$, where $1 < t < n$ and the sets N_i are nonempty, pairwise disjoint and not all of the same cardinality. Let M contain exactly one element from each of the sets N_i. Then there are numbers r and j such that the set $f^r(M)$ contains at least two elements belonging to the set N_j.*

Proof. We assume without loss of generality that the cardinality α of N_1 is maximal among the sets N_i. It follows that $t\alpha > n$. Assume that

$$M = \{a_1, \ldots, a_t\}, \text{ where } a_i \in N_i,\ 1 \le i \le t.$$

Since f is circular, there are nonnegative integers $\beta_i,\ 1 \le i \le t$, such that

$$f^{\beta_i}(a_1) = a_i,\ 1 \le i \le t.$$

(Clearly, we may choose $\beta_1 = 0$.) In the sequel the exponents of f will be reduced modulo n, that is, the smallest nonnegative remainder is always taken.

Since $t\alpha > n$, the sets

$$f^{\beta_i}(N_1),\ 1 \le i \le t,$$

cannot be pairwise disjoint. Consequently, there are integers γ and δ, where $1 \le \gamma < \delta \le t$, such that the sets $f^{\beta_\gamma}(N_1)$ and $f^{\beta_\delta}(N_1)$ have a nonempty intersection. This implies that also the sets N_1 and $f^{\beta_\delta - \beta_\gamma}(N_1)$ have a nonempty intersection. Thus, there is an element $a \in N_1$ such that

$$f^{\beta_\gamma - \beta_\delta}(a) = b \in N_1.$$

Choose a number r_1 such that $f^{r_1}(a_\gamma) = a$. We claim that the choices $r = r_1 + \beta_\gamma - \beta_\delta$ and $j = 1$ satisfy the lemma: f^r maps both a_γ and a_δ to N_1.

Observe first that

$$f^{\beta_\gamma - \beta_\delta}(a_\delta) = a_\gamma.$$

We now obtain

$$f^r(a_\gamma) = f^{\beta_\gamma - \beta_\delta}(f^{r_1}(a_\gamma)) = f^{\beta_\gamma - \beta_\delta}(a) = b \in N_1,$$

as well as

$$f^r(a_\delta) = f^{r_1}(f^{\beta_\gamma - \beta_\delta}(a_\delta)) = f^{r_1}(a_\gamma) = a \in N_1.$$

This proves the claim and the lemma.

If n is prime, the assumption of the sets not being of the same cardinality is always satisfied.

Theorem 5 *Assume that n is prime, f is a circular permutation and g is a function of genus less than n. Then f and g generate all constants.*

Proof. Since f is circular, it suffices to prove that some function of genus 1 is generated. If g is of genus 1, there is nothing to prove. We assume, inductively, that a function h of genus t, where $1 < t < n$, has already been generated and claim that a function of genus $u < t$ can be generated.

Let $b_1, \ldots, b_t$ be the values assumed by h, and let $N_1, \ldots, N_t$ be maximal subsets of the set N satisfying the condition $h(N_i) = b_i$, for $i = 1, \ldots, t$. Since n is prime, the sets N_i cannot all be of the same cardinality. If h^2 is of genus smaller than t, we have established the claim. Otherwise, the numbers b_i are in different sets N_i. Choose now

$$M = \{b_1, \ldots, b_t\}.$$

Choose, further, r and j according to Lemma 6. Then the function hf^rh is of genus $u < t$.

5 Necessary conditions for synchronizability

A general method for obtaining non-synchronizable automata is to investigate properties formulated as follows.

Principle: Property $\mathcal{P}$ is preserved under compositions, and no constant possesses $\mathcal{P}$.

If each function in a given set $\mathbf{F}$ possesses such a property $\mathcal{P}$, then $\mathbf{F}$ is not in SYNCHRO.

One way of finding such properties $\mathcal{P}$ is to consider *self-conjugate* functions. The latter have been widely studied in many-valued logic, where most of the interesting functions have several variables. Therefore, the next definition is stated for functions with *several* variables. The variables still range over the set N, and also the function values are in N.

Definition 4 *A function $f(x_1, \ldots, x_k)$ is* self-conjugate *under the permutation g if*

$$f(x_1, \ldots, x_k) = g(f(g^{-1}(x_1), \ldots, g^{-1}(x_k))).$$

For unary functions, self-conjugacy means simply that the function and the permutation commute: $gf = fg$.

Self-conjugacy constitutes a property $\mathcal{P}$, as defined above, provided the permutation in question has no fixed-points

Lemma 7 *A set of functions is not in* SYNCHRO *if every function in the set is self-conjugate under a permutation g and, moreover, g maps no element of N into itself.*

Proof. Since g commutes with every function in $\mathbf{F}$, it commutes also with every function in $\mathbf{G(F)}$. On the other hand, g cannot commute with any constant i because, by the assumption, g maps i to $j \neq i$.

Using Lemma 7, it is easy to give examples of sets in PRESYNCHRO that contain circular permutations but are not in SYNCHRO, provided n is a composite number. For instance, consider the following automaton with the state set $\{1, 2, 3, 4\}$, input letters a, b, c, d and the transition function defined by the following table:

δ	1	2	3	4
a	2	3	4	1
b	1	4	3	2
c	1	3	3	1
d	4	2	2	4

The automaton is *not synchronizable.* Indeed, each of the functions defined by an input letter is self-conjugate under the permutation (13)(24), and the function induced by the letter a is a circular permutation.

6 Universal synchronizing words

In this final section we will investigate words that will synchronize any synchronizable automaton over a fixed alphabet Σ. Such words do not exist unless we have an upper bound for the cardinality of the state set. For instance, no word over the alphabet $\{a\}$ synchronizes all synchronizable automata over $\{a\}$.

This leads to the following definition.

Definition 5 *A word $w \in \Sigma^*$ is* (universal) K-synchronizing *if, whenever $\mathcal{A} = (Q, \Sigma, \delta)$, $\text{card}(Q) \leq K$ is synchronizable, then w is synchronizing for $\mathcal{A}$.*

We introduce next a related, somewhat more general notion. We do not want to reduce the state set of an automaton to one state (as in connection with a synchronizing word) but to reduce it by a fixed amount. The reduction has to take place independently of the cardinality of the original state set.

Definition 6 *A word w is* K-reducing *for a finite deterministic automaton $\mathcal{A} = (Q, \Sigma, \delta)$ if*

$$\text{card}(\delta(Q, w)) \leq \text{card}(Q) - K.$$

The automaton $\mathcal{A}$ is K-reducible *if it has a K-reducing word. A word $w \in \Sigma^*$ is* (universal) K-reducing *if, whenever $\mathcal{A} = (Q, \Sigma, \delta)$ is K-reducible, then w is K-reducing for $\mathcal{A}$.*

Some facts are immediate consequences of the definitions.

- If $\Sigma = \{a_1, \ldots, a_k\}$ then $a_1 \ldots a_k$ is universal 1-reducing.

- Every universal K-reducing word is universal $(K+1)$-synchronizing.
- For every Σ and K, universal K-synchronizing words exist. This follows because there are only finitely many finite deterministic automata over Σ possessing at most K states. Choose a synchronizing word for each synchronizable automaton among them. Any catenation of such words is universal K-synchronizing over Σ.

(Universal) K-reducing words are called *K-collapsing* in [2]. We do not discuss here the problem of their existence for an arbitrary K. (See [2] and the further references given therein.)

¿From now on we consider alphabets with two letters and 2-reducing, as well as 3-synchronizing words.

Lemma 8 *Consider* $\mathbf{F} = \{f, g\}$*, where* f *and* g *are of genus* $< n$*, such that* $\mathbf{G}(\mathbf{F})$ *contains a function of genus* $\leq n-2$*. Then a composition of length 2, that is, one of the functions* $f^2,\ g^2,\ fg,\ gf$ *is of genus* $\leq n-2$*.*

Proof. We assume without loss of generality that both f and g are of genus exactly $n-1$. For $1 \leq a \leq n$, we denote by N_{-a} the set $\{1, \ldots, n\} - a$. Thus, for some a and b, where possibly $a = b$, we have

$$f(N) = N_{-a},\ g(N) = N_{-b}.$$

We may assume further that

$$f(N_{-a}) = N_{-a},\ g(N_{-b}) = N_{-b},$$

because if this is not the case, then either f^2 or g^2 is of genus $\leq n-2$.

Consider now the set $f(N_{-b})$. It is either of cardinality $\leq n-2$, or else $f(N_{-b}) = N_{-a}$. In the former case we conclude that the function gf is of genus $\leq n-2$. Considering the set $g(N_{-a})$, we conclude similarly that either the function fg is of genus $\leq n-2$, or else $g(N_{-a}) = N_{-b}$.

Thus, we are left with the alternative

$$f(N_{-b}) = N_{-a},\ g(N_{-a}) = N_{-b}.$$

(This alternative necessarily holds if $a = b$.) This implies that the range of any composition of f and g equals either N_{-a} or N_{-b}. Hence, no function in $\mathbf{G}(\mathbf{F})$ is of genus $\leq n-2$, which is a contradiction.

Lemma 9 *Assume that a word* $w \in \{f, g\}^*$ *contains each of the words*

$$fgf,\ fg^2f,\ gfg,\ gf^2g$$

as a subword. Then w *is universal 2-reducing.*

Proof. Consider a 2-reducible automaton with the input alphabet $\{f, g\}$. The functions induced by f and g cannot both be permutations. If they are both of genus $< n$, the lemma is a consequence of Lemma 8. Hence, we may assume that one of them is a permutation and the other of genus $< n$. Because of symmetry, we may assume that (the function induced by) g is a permutation. If f or f^2 is of genus $\leq n-2$, there is nothing further to prove. Thus, we assume that, for some a,

$$f(N) = N_{-a} = f(N_{-a}).$$

We know also that, for some distinct b and c, $f(b) = f(c)$ and, moreover, that one of b and c, say b, is not an element of N_{-a}, that is, $b = a$.

The set $g(N_{-a})$ is of cardinality $n-1$. Thus, either $g(N_{-a})$ contains both a and c, or $g(N_{-a}) = N_{-a}$, or else $g(N_{-a}) = N_{-c}$. In the first case fgf is of genus $\leq n-2$. If $g(N_{-a}) = N_{-a}$, no composition of g and f is of genus $\leq n-2$, which is impossible. Consequently, $g(N_{-a}) = N_{-c}$.

Observe that $f(N_{-c}) = N_{-a}$, or else $f(N_{-c})$ is of cardinality $\leq n-2$, implying that fgf is of genus $\leq n-2$. Thus, we assume that the former alternative holds. The alternatives $g(N_{-c}) = N_{-c}$ and $g(N_{-c}) = N_{-a}$ lead to the conclusion that no function of genus $\leq n-2$ is generated by f and g. This leaves the alternative that that both a and c are contained in $g(N_{-c})$. Hence, fg^2f is of genus $\leq n-2$, which concludes the proof.

Theorem 6 *The words*

$$fg^2f^2gfg,\ fgf^2g^2fg,\ fgfg^2f^2g,\ fg^2fgf^2g,$$

as well as their mirror images, are universal 3-synchronizing. No word of length at most 7 is universal 3-synchronizing over the alphabet $\{f, g\}$. The words listed constitute all of the universal 3-synchronizing words of length 8.

Proof. The first sentence is an immediate consequence of Lemma 9. The other claims follow by a simple case analysis. One observes first that no word w, $|w| \leq 8$, can be universal 3-synchronizing if it misses one of the subwords listed in Lemma 9. If w misses the subword fgf (resp. fg^2f), then we define f by the value sequence 223, and g by the value sequence 321 (resp. 231). Now w cannot be synchronizing for the automaton determined by f and g. (Observe that fg^3f (resp. fg^5f) is synchronizing but the resulting total word will be of length > 8, when the symmetry between f and g is taken into account.) Finally, there is no word of length ≤ 7 having the words listed in Lemma 9 as subwords, and the given words of length 8 are the only ones of length 8 possessing this property.

References

1. R. Adler, I. Goodwin and B. Weiss, Equivalence of topological Markov shifts. Israel J. Math. 27 (1977) 49-63.
2. D.S. Ananichev and M.V. Volkov, Collapsing words vs. synchronizing words. In W. Kuich (ed.) Developments of Language Theory, Preproceedings, TU Wien (2001) 159-170.
3. S. Bogdanović, B. Imreh, M. Ćirić and T. Petković, Directable automata and their generalizations: a survey. Novi Sad J. Math. 29 (1999).
4. J. Černý, Poznámka k homogénnym experimentom s konečnými automatmi. Mat.fyz.čas SAV 14 (1964) 208-215.
5. J. Černý, A. Pirická and B. Rosenauerová, On directable automata. Kybernetika 7 (1971) 289-298.

6. S. Ginsburg, On the length of the smallest uniform experiment which distinguishes the terminal states of a machine. J. ACM 5 (1958) 266–280.
7. J. Kari, A counter example to a conjecture concerning synchronizing words in finite automata. EATCS Bulletin 73 (2001) 146.
8. A. Mateescu and A. Salomaa, Many-valued truth functions, Černý's conjecture and road coloring. EATCS Bulletin 68 (1999) 134-150.
9. E.F. Moore, Gedanken experiments on sequential machines. In C.E. Shannon and J. McCarthy (ed.) Automata Studies, Princeton University Press (1956) 129-153.
10. S. Piccard, Sur les bases du groupe symétrique et les couples de substitutions qui engendrent un goupe régulier. Librairie Vuibert, Paris (1946).
11. J.-E. Pin, Le problème de la synchronisation, Contribution a l'étude de la conjecture de Černý. Thèse de 3^e cycle a l'Université Pierre et Marie Curie (Paris 6) (1978).
12. A. Salomaa, A theorem concerning the composition of functions of several variables ranging over a finite set. J. Symb. Logic 25 (1960) 203-208.
13. A. Salomaa, On the composition of functions of several variables ranging over a finite set. Ann. Univ. Turkuensis, Ser. AI, 41 (1960).
14. A. Salomaa, On basic groups for the set of functions over a finite domain. Ann. Acad. Scient. Fennicae, Ser. AI, 338 (1963).
15. A. Salomaa, Composition sequences for functions over a finite domain. Turku Centre for Computer Science Technical Report 332 (2000), to appear in Theoretical Computer Science.
16. A. Salomaa, Depth of functional compositions. EATCS Bulletin 71 (2000) 143-150.
17. A. Salomaa, Compositions over a finite domain: from completeness to synchronizable automata. In A. Salomaa, D. Wood and S. Yu (eds.) A Half-Century of Automata Theory. Celebration and Inspiration, World Scientific Publ. Co. (2001) 131-143.
18. S.V. Yablonskii, Functional constructions in k-valued logic (in Russian). Tr.Matem.Inst. im. V.A.Steklova 51,5 (1958) 5-142.