Discrete Applied Mathematics 12 (1985) 13-20
North-Holland

THE BASE OF THE INTERSECTION OF TWO FREE SUBMONOIDS*

Ronald V. BOOK
Department of Mathematics, University of California at Santa Barbara, Santa Barbara, CA 93106, USA

Received 13 February 1984
Revised 18 December 1984

Given two free submonoids of a free monoid, one wishes to find a specification for the base of the intersection. An algorithm to construct a graph-theoretic specification of the base is presented. From this specification it can easily be determined whether the base is finite. In addition, a polynomial-time algorithm to determine if a regular set is a circular code is presented.

1. Introduction

It is known [6] that the intersection of two finitely generated subgroups of a free group is again finitely generated. Recently, Avenhaus and Madlener [1] have given a polynomial time algorithm for computing a set of generators for such an intersection. In the case of monoids, the situation is different. First, a submonoid of a free monoid is not necessarily free. Second, while the intersection of two finitely generated free submonoids is again free, this intersection is not necessarily finitely generated.

It is well known that it is decidable whether the submonoid of a free monoid which is generated by a finite set (or even a regular set) is free. The purpose of this paper is to present a polynomial time algorithm to generate a graph-theoretic specification of the base of the intersection of two free submonoids that are regular sets. Each submonoid is specified by a deterministic finite-state acceptor for its base. The running time of the algorithm is linear in the product of the sizes of the transition tables for the two finite-state acceptors. From this specification of the base of the intersection, it is easy to determine whether the base is finite.

This algorithm is a minor modification of an algorithm due to Even [4,5] that is useful in the study of variable-length codes. Comments on Even's original work are given in Section 3 and circular codes are considered. A polynomial-time algorithm to determine whether a regular set is a circular code is presented.

*This research was supported in part by the National Science Foundation under Grants MCS81-16327 and DCR83-14977.

1. Submonoids of a free monoid

For a finite set Σ of symbols, let Σ^* denote the free monoid generated by Σ. If $A \subseteq \Sigma^*$, then let A^* denote the submonoid of Σ^* generated by A, that is, $A^* = \{y_{i_1} \cdots y_{u_n} \mid n \geq 1, \text{ each } y_{i_j} \in A\} \cup \{e\}$ where e is the empty string, the string of length 0. For every $A \subseteq \Sigma^*$, there is a unique minimal generating set $B_A \subseteq \Sigma^*$ such that $B_A^* = A^*$; the set B_A is $A^+ - A^+A^+$, where $A^+ = A^* - \{e\}$, and is called the *base* of A^*. Recall that the submonoid A^* is free if and only if every string in A^* has a unique factorization as a concatenation of strings in B_A.

We assume that the reader is familiar with the fundamentals of the theory of regular sets, for example, that a subset of Σ^* is regular if and only if it is recognized by a deterministic or nondeterministic finite-state acceptor if and only if it is denoted by a regular expression. Recall that the class of regular subsets of Σ^* is the smallest class containing the finite subsets and closed under union, concatenation, and *.

It is easy to see that there is a nondeterministic procedure that on input a regular set $A \subseteq \Sigma^*$ (say A is specified by a finite-state acceptor) will determine whether A is not a base, and that operates within $\log n$ work space. Hence, there is a deterministic polynomial time algorithm to determine whether a regular set is a base. Further, the algorithm of Even [4,5,3] that determines whether A^* is free when A is a finite base can be extended to regular sets, and this extension again operates deterministically in polynomial time.

Now it is clear that if A^* and B^* are free submonoids of Σ^*, then so is $A^* \cap B^*$, and it is well known that if A^* and B^* are regular sets, then so are $A^* \cap B^*$ and the base of $A^* \cap B^*$. On the other hand, if A and B are finite, then the base of $A^* \cap B^*$ is not necessarily finite. For example, let $\Sigma = \{a, b\}$, $A = \{a, ab\}$, and $B = \{a, ba\}$. It is clear that A^* and B^* are free. Let $C = \{(ab)^i a \mid i \geq 0\}$. Clearly, C is a base and C^* is free. We claim that $C^* = A^* \cap B^*$ so that $A^* \cap B^*$ is not finitely generated. For any i, $(ab)^i a$ is in A^* and $a(ba)^i$ is in B^* so that $(ab)^i a \in A^* \cap B^*$; hence, $C^* \subseteq A^* \cap B^*$. To show that $A^* \cap B^* \subseteq C^*$, consider the base of $A^* \cap B^*$, say D, and argue by induction on the length of strings in D that $D \subseteq C$, concluding $D = C$; the details are left to the reader.

2. Computing a specification of the basis

Theorem 1. *Consider the following problem:*

INSTANCE: *Two deterministic finite-state acceptors M_A and M_B specifying the regular sets A and B respectively, where A is a base, A^* is free, B is a base, and B^* is free.*

QUESTION: *Is the base of $A^* \cap B^*$ finite?*

There is an algorithm to construct a labeled directed graph to specify the base of $A^* \cap B^*$. This algorithm runs in time $O(|\delta_A| \cdot |\delta_B|)$ where $|\delta_A|$ ($|\delta_B|$) is the size of the

table specifying the transition function of M_A *(resp.,* M_B*). From the graph constructed by this algorithm, one can determine whether the base of* $A^* \cap B^*$ *is finite in time* $O(|\delta_A|^2 \cdot |\delta_B|^2)$.

The remainder of this section is devoted to the proof of Theorem 1. It will be assumed throughout that A and B are regular sets. The other portions of the hypothesis will be explicitly stated whenever they are used.

Let A be a regular set and let $M_A = (K_A, \Sigma, \delta_A, q_A, F_A)$ be a deterministic finite-state acceptor that recognizes A. (That is, K_A is a finite set of states, $\delta_A : K_A \times \Sigma \to K_A$ is the transition function, $q_A \in K_A$ is the initial state, and $F_A \subseteq K_A$ is the set of accepting states; δ_A is extended to $\delta_A^* : K_A \times \Sigma^* \to K_A$ by $\delta_A^*(q, e) = q$ and $\delta_A^*(q, wa) = \delta_A(\delta_A^*(q, w), a)$ for all $q \in K_A$, $a \in \Sigma$, and $w \in \Sigma^*$; the set of strings recognized by M_A is $\{w \in \Sigma^* \mid \delta_A^*(q_A, w) \in F_A\}$.) Construct the following labelled directed graph $G_A = (K_A \cup \{s_A, f_A\}, \Sigma, \delta(A), \lambda_A, f_A)$ as follows:

(a) The set of vertices is $K_A \cup \{s_A, f_A\}$, where $s_A \neq f_A$ and $\{s_A, f_A\} \cap K_A = \emptyset$.

(b) The set of directed edges is $\delta(A) = \{(p, q) \mid p, q \in K_A$ and for some $a \in \Sigma$, $\delta_A(p, a) = q\} \cup \{(s_A, q) \mid$ for some $a \in \Sigma$, $\delta_A(q_0, a) = q\} \cup \{(p, f_A) \mid$ for some $a \in \Sigma$, $\delta_A(p, a) \in F_A\} \cup \{(s_A, f_A) \mid$ for some $a \in \Sigma$, $\delta_A(q_0, a) \in F_A\}$.

(c) The labelling function is λ_A: $(K_A \cup \{s, f\}) \times (K_A \cup \{s, f\}) \to 2^\Sigma$ where for every $p, q \in K_A$, $a \in \Sigma$,

(i) $a \in \lambda_A(p, q)$ if and only if $\delta_A(p, a) = q$,
(ii) $a \in \lambda_A(p, f_A)$ if and only if $\delta_A(p, a) \in F_A$,
(iii) $a \in \lambda_A(s_A, q)$ if and only if $\delta_A(q_0, a) = q$,
(iv) $a \in \lambda_A(s_A, f_A)$ if and only if $\delta_A(q_0, a) \in F_A$.

The labelled directed graph G_A is a modification of the state graph of M_A. The vertex s_A has in-degree zero and is a 'copy' of q_A. The vertex f_A has out-degree zero and 'copies' all of the states in F_A simultaneously. Since M_A is deterministic, for each $w \in \Sigma^*$, there is a unique computation of M_A on w that begins at q_A. It is easy to see from the construction of G_A that for every $w \in A$, there is a unique sequence $p_0, p_1, \ldots, p_{|w|}$ of vertices in G_A such that $p_0 = s_A$, $p_{|w|} = f_A$, and for each $i = 1, \ldots, |w|$, $a_i \in \lambda_A(p_{i-1}, p_i)$ where $w = a_1 \ldots a_{|w|}$. Since A is a base, $e \notin A$.

The *coding graph* $C(A) = (K_A \cup \{s_A\}, \Sigma, \delta^A, \lambda^A, s_A, s_A)$ is obtained from the graph G_A as follows:

(a) The set of vertices is $K_A \cup \{s_A\}$.

(b) The set of directed edges is $\delta^A = (\delta(A) - \{(p, f_A) \mid p \in K_A \cup \{s_A\}\}) \cup \{(p, s_A) \mid p \in K_A \cup \{s_A\}$ and $(p, f_A) \in \delta(A)\}$.

(c) The labelling function is $\lambda^A : (K_A \cup \{s_A\}) \times (K_A \cup \{s_A\}) \to 2^\Sigma$ where for $p \in K_A \cup \{s_A\}$ and $q \in K_A$, $\lambda^A(p, q) = \lambda_A(p, q)$, and for $p \in K_A \cup \{s_A\}$, $\lambda^A(p, s_A) = \lambda_A(p, f_A)$.

The coding graph $C(A)$ is obtained from G_A by identifying s_A and f_A. Clearly, for $n \geq 1$ and $a_1, \ldots, a_n \in \Sigma$, $a_1 \ldots a_n \in A^*$ if and only if there is a sequence p_0, $p_1, \ldots, p_n \in K_A \cup \{s_A\}$ such that $p_0 = p_n = s_A$ and for each $i = 1, \ldots, n$, $a_i \in \lambda^A(p_{i-1}, p_i)$; the sequence $p_0, p_1, \ldots, p_n$ is a *path-sequence that witnesses* $a_1 \ldots a_n \in A^*$.

Claim 1. *Suppose that A is a base and A^* is free. Then, for $n \geq 1$ and $a_1, \ldots, a_n \in \Sigma$, $a_1 \ldots a_n \in A^*$ if and only if in $C(A)$ there is a unique path-sequence that witnesses $a_1 \ldots a_n \in A^*$.*

This fact follows immediately from the definitions and the construction of G_A and of $C(A)$.

Now we construct a 'testing graph' that provides the basis for our result.

For regular sets A and B, let $G_A = (K_A \cup \{s_A, f_A\}, \Sigma, \lambda_A, \delta(A), s_A, f_A)$ and $G_B = (K_B \cup \{s_B, f_B\}; \Sigma, \lambda_B, \delta(B), s_B, f_B)$ be the corresponding labelled directed graphs as described in the last paragraph. Define the *testing graph* $T(G_A, G_B) = (V, E, \lambda)$, where V is the set of vertices, E is the set of edges, and $\lambda : E \to 2^{\Sigma}$ is the labelling function, as follows:

(a) $V = ((K_A \cup \{s_A\}) \times (K_B \cup \{s_B\})) \cup \{(f_A, f_B)\}$.

(b) E is defined as follows: let $p_1 \in K_A \cup \{s_A\}$ and $p_2 \in K_B \cup \{s_B\}$,

(i) for each $q_1 \in K_A$ and $q_2 \in K_B$, there is an edge from (p_1, p_2) to (q_1, q_2) if and only if $\lambda_A(p_1, q_1) \cap \lambda_B(p_2, q_2) \neq \emptyset$;

(ii) for each $q_2 \in K_B$, there is an edge from (p_1, p_2) to (s_A, q_2) if and only if $\lambda_A(p_1, f_A) \cap \lambda_B(p_2, q_2) \neq \emptyset$;

(iii) for each $q_1 \in K_A$, there is an edge from (p_1, p_2) to (q_1, s_B) if and only if $\lambda_A(p_1, q_1) \cap \lambda_B(p_2, f_B) \neq \emptyset$;

(iv) there is an edge from (p_1, p_2) to (f_A, f_B) if and only if $\lambda_A(p_1, f_A) \cap \lambda_B(p_2, f_B) \neq \emptyset$;

(c) The definition of λ parallels (b):

(i) $\lambda((p_1, p_2), (q_1, q_2)) = \lambda_A(p_1, q_1) \cap \lambda_B(p_2, q_2)$,

(ii) $\lambda((p_1, p_2), (s_A, q_2)) = \lambda_A(p_1, f_A) \cap \lambda_B(p_2, q_2)$,

(iii) $\lambda((p_1, p_2), (q_1, s_B)) = \lambda_A(p_1, q_1) \cap \lambda_B(p_2, f_B)$,

(iv) $\lambda((p_1, p_2), (f_A, f_B)) = \lambda_A(p_1, f_A) \cap \lambda_B(p_2, f_B)$.

It is clear from the definition of $T(G_A : G_B)$ that the in-degree of (s_A, s_B) and the out-degree of (f_A, f_B) are both zero.

The set $W(A, B)$ of *word paths* of $T(G_A, G_B)$ is the set of strings $a_1 \ldots a_n$, $n \geq 1$, such that there is a sequence $r_0, r_1, \ldots, r_n$, $r_0 = (s_A, s_B)$, $r_n = (f_A, f_B)$, where $r_i \in (K_A \cup \{s_A\}) \times (K_B \cup \{s_B\})$ for $0 < i < n$, and for each $i = 1, \ldots, n$, $a_i \in \Sigma$ and $a_i \in \lambda(r_{i-1}, r_i)$; the sequence $r_0, r_1, \ldots, r_n$ is a *path-sequence that witnesses* $a_1 \ldots a_n \in W(A, B)$.

Let $r_0, r_1, \ldots, r_n$ witness $a_1 \ldots a_n \in W(A, B)$. For each $i = 1, \ldots, n-1$, let $r_i = (p_i, q_i)$

where $p_i \in K_A \cup \{s_A\}$ and $q_i \in K_B \cup \{s_B\}$. It is clear that the sequence $S_A, p_1, \ldots, p_{n-1}, s_A$ in $C(A)$ witnesses $a_1 \ldots a_n \in A^*$ and the sequence $s_B, q_1, \ldots, q_{n-1}, s_B$ in $C(B)$ witnesses $a_1 \ldots a_n \in B^*$. Thus, $W(A, B) \subseteq A^* \cap B^*$.

Claim 2. $W(A, B)^* = A^* \cap B^*$.

Proof. Since $W(A, B) \subseteq A^* \cap B^*$ and $A^* \cap B^* = (A^* \cap B^*)^*$, it is sufficient to show that if D is the base of $A^* \cap B^*$, then $D \subseteq W(A, B)^*$. In fact, we show that $D \subseteq W(A, B)$.

Let $a_1 \ldots a_n \in D$ where $n \geq 1$ and each a_i is in Σ. Since D is the base of $A^* \cap B^*$, $a_1 \ldots a_n$ is in $A^* \cap B^*$ and $a_1 \ldots a_n$ has no factorization as the concatenation of two or more nonempty strings in $A^* \cap B^*$. Hence, there is a path-sequence $p_0, p_1, \ldots, p_n$ in $C(A)$ witnessing $a_1 \ldots a_n \in A^*$ and a path-sequence $q_0, q_1, \ldots, q_n$ in $C(B)$ witnessing $a_1, \ldots, a_n \in B^*$, so that $p_0 = p_n = s_A$ and $q_0 = q_n = s_B$, and for each $i = 1, \ldots, n-1$, either $p_i \neq s_A$ or $q_i \neq s_B$ (for otherwise, $a_1 \ldots a_n$ would have a factorization as the concatenation of two or more nonempty strings in $A^* \cap B^*$). Let $r_0 = (p_0, q_0)$ $(= (s^A, s_B))$, $r_n = (f_A, f_B)$, and for each $i = 1, \ldots, n-1$, $r_i = (p_i, q_i)$; then it is clear that $r_0, r_1, \ldots, r_n$ witnesses $a_1 \ldots a_n \in W(A, B)$. □

In general, $W(A, B)$ is not a base: let $A = \{ababab, aba, bab, aaa\}$ and $B = \{ababab, aba, bab, bbb\}$, so that $\{ababab, aba, bab\} \subseteq W(A, B)$.

Claim 3. *Suppose that A is a base, A^* is free, B is a base, and B^* is free. Then $W(A, B)$ is the base of $A^* \cap B^*$.*

Proof. From the proof of Claim 2, we have $D \subseteq W(A, B)$ where D is the base of $A^* \cap B^*$. Since $W(A, B) \subseteq A^* \cap B^*$, $D^* = W(A, B)^* = A^* \cap B^*$ so that it is sufficient to show that $W(A, B)$ is a base.

Assume that $W(A, B)$ is not a base. Let $a_1 \ldots a_n$, $n \geq 1$, each $a_i \in \Sigma$, be a string in $W(A, B)$ with a factorization as the concatenation of two or more strings in $W(A, B)$, say $a_1 \ldots a_n = w_1 \ldots w_k$ with $k > 1$ and $w_i \in W(A, B)$ for each $i = 1, \ldots, k$. For each $i = 0, 1, \ldots, n$, let $r_i = (p_i, q_i)$ where $r_0, \ldots, r_n$ is a path-sequence in $T(G_A, G_B)$ witnessing $a_1 \ldots a_n \in W(A, B)$. By construction of $T(G_A, G_B)$, $r_i \neq (S_A, S_B)$ if $i \neq 0$. Since A is a base and A^* is free, there is a unique path-sequence in $C(A)$ witnessing $a_1 \ldots a_n \in A^*$ and by construction this sequence is $s_A, p_1, \ldots, p_{n-1}, s^a$. Also, since B is a base and B^* is free, there is a unique path-sequence in $C(B)$ witnessing $a_1 \ldots a_n \in B^*$ and by construction this sequence is $s_B, q_1, \ldots, q_{n-1}, s_B$. Now each w_i is in $A^* \cap B^*$ and so for each w_i there is a path-sequence in $C(A)$ witnessing $w_i \in A^*$ and a path-sequence in $C(B)$ witnessing $w_i \in B^*$. From this collection of path-sequences, we can construct a path-sequence $\overline{p_0}, \ldots, \overline{p_n}$ in $C(A)$ witnessing $w_1 \ldots w_k \in A^*$ and a path-sequence $\overline{q_0}, \ldots, \overline{q_n}$ in $C(B)$ witnessing $w_1 \ldots w_k \in B^*$, where $\overline{p_0} = \overline{p_n} = s_A$ and $\overline{q_0} = \overline{q_n} = s_B$. Since $w_1 \ldots w_k$ is the concatenation of k strings in $A^* \cap B^*$, there are $k-1$ integers j in $\{1, \ldots, n-1\}$ such that $(\overline{p_j}, \overline{q_j}) = (s_A, s_B)$.

Thus, the sequence $(p_0, q_0), \ldots, (p_{n-1}, q_{n-1}), (s_A, s_B)$ is not equal to the sequence $(\overline{p_0}, \overline{q_0}), \ldots, (\overline{p_n}, \overline{q_n})$. Since $a_1 \ldots a_n = w_1 \ldots w_k$, this means that there are two or more path-sequences in $C(A)$ witnessing $a_1 \ldots a_n \in A^*$ or there are two or more path-sequences in $C(B)$ witnessing $a_1 \ldots a_n \in B^*$, a contradiction. □

We are concerned with the cardinality of $W(A, B)$.

Claim 4. *$W(A, B)$ is infinite if and only if there are infinitely many paths from (s_A, s_B) to (f_A, f_B) in $T(G_A, G_B)$ if and only if there is a path from (s_A, s_B) to (f_A, f_B) in $T(G_A, G_B)$ that contains a circuit if and only if there is a path from (s_A, s_B) to (f_A, f_B) in $T(G_A, G_B)$ of length greater than $m-1$ and less than $2m-1$ where m is the number of vertices in $T(G_A, G_B)$.*

It is clear that the graph $T(G_A, G_B)$ can be constructed from M_A and M_B in time $O(|\delta_A| \cdot |\delta_B|)$, where $|\delta_A|$ ($|\delta_B|$) is the size of the table specifying the transition function of M_A (resp., M_B). Breadth-first search can be used to determine whether $W(A, B)$ is infinite and this can be done in time $O(|\delta_A|^2 \cdot |\delta_B|^2)$. Hence, there is a polynomial time algorithm that on input M_A and M_B will determine whether $W(A, B)$ is infinite.

Notice that if A is a base, A^* is free, B is a base, and B^* is free, then a specification by means of a finite labelled directed graph is provided for the base of $A^* \cap B^*$ by pruning from $T(G_A, G_B)$ those vertices which do not lie on a path from (s_A, s_B) to (f_A, f_B).

3. Even's algorithm and circular codes

Even's original work [4,5] was to describe a graph-theoretic algorithm to determine whether a finite set $A \subset \Sigma^*$ is such that A^* is free on A, that is, A is a base and A^* is free. This was presented in the context of A being a variable length code so that A is a base and A^* is free if and only if that A is a uniquely decipherable code. In the original work A was taken as a finite set; here we describe the construction for the extension to the case of A being a regular set such that $e \notin A$.

Let M_A be a deterministic finite-state acceptor recognizing regular set A, and let G_A and $C(A)$ $(= (K_A \cup \{s_A\}, \Sigma, \delta^A, \lambda^A, s_A, s_A)$ be the graphs described in Section 2. Construct the testing graph $T(A)$ of A as follows:

(i) The set of vertices is $(K_A \cup \{s_A\}) \times (K_A \cup \{s_A\}) \cup \{I\}$, where I is not in $(K_A \cup \{s_A\}) \times (K_A \cup \{s_A\})$.

(ii) The set of edges is defined inductively as follows:

if $p, q \in K_A \cup \{s_A\}$ and for some $a \in \Sigma$, $a \in \lambda^A(s_A, p) \cap \lambda^A(s_A, q)$, then there is an edge from I to (p, q);

if $p, q, r, s \in K_A \cup \{s_A\}$, there is a path from I to (p, q), and for some $a \in \Sigma$, $a \in \lambda^A(p, r) \cap \lambda^A(q, s)$, then there is an edge from (p, q) to (r, s).

Then A is a base and A^* is free if and only if there is no $p \in K_A$ such that there exist a path in $T(A)$ from I to (p, s_A) and a path in $T(A)$ from (p, s_A) to (s_A, s_A).

By generating the pairs (p, s_A), $p \neq s_A$, that are accessible from I and then determining whether (s_A, s_A) is accessible from such a pair, one can determine whether A^* is free. We summarize in the following way.

Theorem 2. *Consider the following problem:*

INSTANCE: *A deterministic finite-state acceptor with n states recognizing a regular set A such that A is a base.*

QUESTION: *Is A^* free?*

There is an algorithm that will solve this problem in time at most $O(n^4)$.

Now we consider an application of these techniques to the question of whether a uniquely decipherable code is circular.

A set $A \subset \Sigma^*$ is a *circular* code if A is uniquely decipherable and for all choices of $x_1, \ldots, x_n, y_1, \ldots, y_m \in A$ and $u, v \in \Sigma^*$, $v \neq e$, if $vx_2 \ldots x_n u = y_1 \ldots y_m$ and $x_1 = uv$, then $u = e$.

We will describe a polynomial-time algorithm that on input a deterministic finite-state acceptor M_A recognizing regular set A will determine whether A is a circular code. From the last result we can assume that it is already known that A is uniquely decipherable.

Let $C(A) = (K_A \cup \{s_A\}, \Sigma, \delta^A, \lambda^A, s_A, s_A)$ be the graph described in Section 2. For $p \in K_A$, let $C_p(A)$ be the graph $(K_A \cup \{s_A\}, \Sigma, \delta^A, \lambda^A, p, p)$. Let T_p be the graph $T(C(A), C_p(A))$ as described in Section 2.

Now A is a circular code if and only if for every $p \in K_A$, there is no nonempty path from (s_A, p) to $(s_A, p\&$ in T_p.

This yields the following result.

Theorem 3. *Consider the following problem:*

INSTANCE: *A deterministic finite-state acceptor with n states recognizing a regular set A such that A is a base and A^* is free.*

QUESTION: *Is A a circular code?*

There is an algorithm that will solve this problem in time at most $O(n^5)$.

The bound of $O(n^5)$ results from the fact that T_p has n^2 states so determining whether T_p has no path from (s_A, p) to (s_A, p) requires time at most $O(n^4)$. There are n states and T_p must be checked for each choice of p.

Acknowledgment

I am happy to thank Dr. Craig Squier for some stimulating conversations on this topic. Dr. Friedrich Otto observed that the original 'proof' of Theorem 2 was not correct and provided the necessary alteration.

References

[1] J. Avenhaus and K. Madlener, How to compute generators for the intersection of subgroups in free groups, CAAP '81, Lecture Notes in Computer Science 112 (1981) 88–100.

[2] J. Berstel and D. Perrin, Codes circulaires, in: L. Cummings, ed., Combinatorics on Words: Progress and Perspectives (Academic Press, New York, 1983) 133–165.

[3] R. Book, S. Even, S. Greibach, and G. Ott, Ambiguity in graphs and expressions, IEEE Trans. Computers 20 (1971) 149–153.

[4] S. Even, On Information Lossless Automata, Ph.D. dissertation, Harvard University, 1963.

[5] S. Even, On information lossless automata of finite order, IEEE Trans. Computers 14 (1965) 561–569.

[6] A. Howson, On the intersection of finitely generated free groups, J. London Math. Soc. 29 (1954) 428–434.