Università degli Studi di Palermo
Dipartimento di Matematica e Applicazioni
Dottorato di Ricerca in Matematica
- XVIII ciclo -

Automata-theoretic methods in free monoids and free groups

AUTHOR
Laura Giambruno

COORDINATOR
Prof. Vassil Kanev

RESEARCH SUPERVISOR
Prof. Antonio Restivo

Settore scientifico-disciplinare INF/01

Ringraziamenti

Vorrei ringraziare il mio caro 'boss', il Prof. Antonio Restivo per tutto quello che mi ha insegnato e per la sua costante presenza. Egli è stato sempre disponibile ad ascoltare i miei dubbi, le mie perplessità anche in periodi balneari quando i controesempi venivano fuori nei momenti e luoghi meno opportuni e che con la sua ironia ha spezzato, lasciandomi qualche volta 'con le ginocchia per terra', la mia drammaticità nell'affrontare i cari 'automi ribelli alle nostre formule'.

Vorrei ringraziare 'le professeur' Pascal Weil per tutto quello che mi ha insegnato, che esso sia il francese, l'essere puntigliosa e precisa o la teoria dei gruppi rivista avec les automates. Lo vorrei ringraziare anche per la sua disponibilità e gentilezza.

Last but not least vorrei ringraziare Giuseppina Rindone, colei che mi ha sorretto e aiutato nella scrittura della tesi. Con la sua pazienza e la sua simpatia mi ha dato un supporto, fiducia in me stessa e forza per scriverla nonché notevole aiuto matematico. La ringrazio per i suggerimenti che mi ha dato che hanno portato al secondo algoritmo per trovare una base strongly Nielsen reduced di un sottogruppo fissato.

Vorrei ringraziare le colleghe e il collega della mia stanza che hanno sopportato in silenzio i miei nervosismi e mi hanno supportato con la loro allegria: Alessandra, Gaetana, Giusi, Alfonso, Daniela.

Contents

Introduction

Automata-theoretic tools have been deeply utilized in literature for studying algebraic matters, in particular free monoids and free groups. In this thesis we propose to study finitely generated submonoids of free monoids and finitely generated subgroups of free groups, and their intersection.

Free monoids play an important role in combinatorics on words and in formal language theory. The study of submonoids of free monoids has been deepened with combinatorial and automata methods in the setting of the theory of variable-length codes, started by M.P. Schützenberger in [28]. A complete treatment of the theory of codes can be found in [2]. For some recent developments see chap. 6 of [19]. Syntactic properties of submonoids of a free monoid have been investigated in [29] in connection with combinatorics on words. Some recent results in this area are in [26]. Moreover the intersection of two finitely generated submonoids has been object of study since Tilson's paper in 1972 ([35]). It has been investigated by several authors as Karhumäki ([16]), Latteux and Leguy ([17]) and Bruyére, Derencourt and Latteux ([5]). The approach mostly used by the authors is purely combinatorial. Instead, we propose another approach that makes use of automata.

For what concerns free groups, several open problems have been solved and moreover several algorithms concerning group's problems have been optimized using automata.

A graphical and powerful approach in studying free groups was introduced by Serre and Stalling in their fundamental papers of 1977 and 1983 respectively ([30], [32]). In particular, they represent subgroups of a free group $F(A)$ using immersions in the bouquet of $|A|$ circles. Using such representation Stallings ([32]) showed how many algorithms concerning subgroups of free group could be simplified.
Such representation can also be viewed in terms of $A-$labelled graphs ([22]) or in terms of inverse automata. This last point of view has been introduced by Reidemister ([25]) in order to give a simple proof of the Nielsen-Schreier theorem, that states that every subgroup of a free group is free. It has been further developed in many papers ([4], [21], [31] and [37]) where it was introduced a correspondence between 'inverse automata' and finitely generated subgroups.

Such methodology has been applied in the search of an upper bound for the rank of the intersection of two finitely generated subgroups. The intersection of two finitely generated subgroups has been studied since Howson's 1954 paper ([13]) that stated

that the intersection of finitely generated subgroups of a free group is finitely generated and gave an upper bound for the rank of the intersection $H \cap K$ in terms of the ranks of H and K.

Many different inequalities were proved during the last fifty years ([23], [24], [6]). In particular, in 1956 Hanna Neumann (cf [23]) proposed a conjecture, known as Hanna Neumann conjecture, that states the following:

$$\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$$

where, for each T subgroup of $F(A)$, $\widetilde{rk}(T) = max(rk(T) - 1, 0)$. Many improvements have been done during these years (cf. [24], [33], [34], [8]) and in 2002 Meakin and Weil ([22]) proved a stronger version of such conjecture for the class of positively generated subgroups of the free group $F(A)$ on A, finite alphabet, that are generated by words on A^*.

On the track of such results we investigate the correspondence between submonoids of a free monoid and automata and we study the intersection of two finitely generated submonoids using automata.

In particular, like in free groups, we associate to each submonoid a 'special' automaton recognizing it and to every such automaton the language recognized by it that is a submonoid. One of the main difference in the behaviour between free monoids and free groups using automata-theoretic tools, is the research of a basis, and of the cardinality of such basis, depending on the characteristics (like number of vertices and edges) of the automata associated to them. Such differences lead to different outcomes in free monoids and in free groups.

In free groups, given an automaton associated to a subgroup, there exists a plain algorithm for finding a basis of the subgroup. However there can be different bases, having the same cardinality, called the rank of the subgroup. So there is the problem of finding a 'better possible' basis. Such a rank is related to the characteristics of the graph of the automaton. On the contrary, the submonoids of a free monoid have a unique basis but there is no easy connection between their cardinality and the characteristics of the graph of the automata associated to them.

Another aim of this thesis is the research of a general formula linking the rank of a submonoid with the characteristics of the automaton associated to it and the research of a 'better' basis in an automaton associated to a subgroup.

The study of free monoids plays an important role in the combinatoric on words. Moreover the study of the intersection of two submonoids of finite rank is very interesting and not trivial at all. In fact, by a result of Latteux and Leguy ([17]), every language is regular if and only if the submonoid generated by it is obtained as omomorphic image of the intersection of two finitely generated monoids:

Let A be an alphabet and R a language of A^*. R is a regular language if and only if there exist two finite languages F_1, F_2 and a morphism g such that $R^* = g({F_1}^* \cap {F_2}^*)$

The study of the intersection of two submonoids has been developed in many papers. It was first studied by Tilson in 1972 ([35]) who proved that the intersection of two free submonoids of A^*, the free monoid on a finite alphabet A, is free too.

Karhumäki in 1984 ([16]) deepened the study of the intersection of two submonoids generated by two elements by giving a characterization of such an intersection. In particular he proved that given two submonoids H and K of A^*, where A is a finite alphabet, if both H and K are of rank two, then $H \cap K$ is a submonoid generated by at most two words or by a regular language of a special form. In particular if H and K are generated by prefix (or suffix) sets of two words, and $H \cap K$ is not finitely generated, then this intersection has the form $(\alpha\beta^*\gamma)^*$ where $\alpha, \beta, \gamma \in A^*$.

Recently Bruyére, Derencourt and Latteux ([5]) have analyzed the *meet* of two rational codes X and Y, defined as the basis of the free monoid $X^* \cap Y^*$. They concentrate on the study of maximal rational codes such that their meet is yet a maximal rational code, showing with many examples the complex behaviour of the meet. Finally they proved that any rational code is the meet of two rational maximal codes.

One of the starting motivations of this thesis has been the paper of Karhumäki ([16]) that characterizes the intersection of two submonoids of the free monoid generated by two elements. The proof given by Karhumäki provides no intuition on the real nature of the result. In this thesis we prove, in particular, the result of Karhumäki in the case of two submonoids generated by prefix (or suffix) sets using automata.

There exists a lot of literature concerning automata recognizing submonoids ([2], [7]). The main example of automaton recognizing a submonoid is the flower automaton, that is in particular an automaton with a unique final state equal to a unique initial state. So we define a *monoidal automaton* as an automaton that has a unique final state equal to a unique initial state and we study the submonoids-monoidal automata correspondence.

Through the study of the product of two automata associated to two finitely generated submonoids H and K, we are able to prove that if H and K are submonoids generated by prefix sets such that the product automaton associated to $H \cap K$ has a given special property then $\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$.

And in the general case we have found a family of examples such that $rk(H \cap K) = 2^{log_2(rk(H))log_2(rk(K))}$.

Moreover if the two submonoids H and K are generated by prefix sets of two elements, then their intersection $H \cap K$ either is of rank two, or it has the form $(uv^*w)^*$ where $u, v, w \in A^*$, that is the result of Karhumäki ([16]).

For what concerns groups, given an inverse automaton corresponding to a subgroup, we associate to it a graph, called the positive state graph. We have that to every spanning tree of such a graph it corresponds a basis of the subgroup.

One can pose the problem of choosing the spanning tree whose associated basis is the 'best possible'. In particular the bases where the cancellation of the elements in the product is reduced to the minimum: such bases are called 'strongly Nielsen reduced bases '.

Such bases are well-known in literature ([14]). Several algorithms ([14] ,[1]) have been written for finding, given a finite set of words U in a free group, an equivalent set V that is a Strongly Nielsen reduced (i.e. a set generating the same subgroup of that one generated by U). In particular, Avenhaus and Madlener in 1984 ([1]) found an algorithm that, given a finite set of m elements in a free group whose maximum length is n, constructs a strongly Nielsen reduced set in time $O(m^5n^2)$.

In particular, given a finite set of words U, we furnish two algorithms for the construction of a strongly Nielsen basis equivalent to U. The first one carries, by means of automata, a Nielsen reduced basis into a strongly Nielsen reduced basis by using techniques similar to the Nielsen transformations, usually defined on words. For the second one we prove that, given an inverse automaton, the spanning tree associated to the breadth-first search in the initial-final vertex of the graph associated to the automaton gives rise to a strongly Nielsen basis. The algorithm that constructs such tree is an algorithm for the strongly Nielsen reduction with a better complexity than that one of Avenhaus and Madlener.

In particular, given a finite set of m elements in a free group whose sum of length is l, our algorithm constructs a strongly Nielsen reduced set in time $O(l\ log^*l)$, where log^*l is the least natural number k such that applying k times iteratively the function log we find a number less than 1.

After a first draft of the thesis we came across a paper of Kapovich and Myasnikov (cf. [15]) who found a family of spanning trees whose associated bases are strongly Nielsen and that include the breadth-first search tree.

The thesis is organized as follows:

In the first chapter we briefly give the basic tools from the settings of algebra and computer science which will be useful for the results in the following chapters. We begin by giving, in sections 1.1 and 1.2, some fundamental algebraic notions in the theory of monoids and theory of groups. Then we give the basic definitions in combinatoric of words (cf. section 1.3) and in the theory of automata (cf. section 1.4). We conclude section 1.4 with some standard definitions and properties on graphs, pointing out the breadth-first procedure for visiting a graph.

In the second chapter we consider a correspondence between submonoids of the free monoid on A and monoidal automata on A. Such a correspondence relates properties of the submonoids with those of the automata. In particular, defining a *semi-flower automaton* as a monoidal automaton in which all cycles visit the initial-final vertex, we get that to finitely generated submonoids correspond to semi-flower automata. In section 2.2, after introducing and investigating the notions of *branch point going out*

and *branch point going in*, in short *bpo* and *bpi*, in an automaton, we study semi-flower automata with a unique bpi. In this case we find a general formula for the rank of the submonoid depending on the characteristic of the automaton. This formula brings us, in section 2.3, to the proof of the Hanna Neumann inequality for product automata with a unique bpi. In 2.4 we study, by using automata, the intersection of two submonoids generated by prefix sets of two words and we obtain a new proof of the result of Karümaki. Finally, in section 2.4, we discuss possible approaches to the other cases.

In the third chapter we recall the automata-theoretic methods in studying subgroups of the free group. In section 3.1 we recall the well-known correspondence between subgroups of the free group on A and inverse automata on A. In section 3.2 we recall the arguments that lead to the Hanna Neumann inequality, underlining the difference with the free monoid's case. In section 3.3 we recall definitions and properties of Nielsen bases and strongly Nielsen bases. Finally, in section 3.4 we give two algorithms for the strongly Nielsen reduction and we discuss the complexity of the second one.

Chapter 1

Preliminaries

In this chapter we briefly give the basic tools from the setting of algebra and computer science which will be useful for the results in the following chapters. We begin by giving, in sections 1.1 and 1.2, some fundamental algebraic notions in the theory of monoids and theory of groups, underlying the different behaviour of these algebraic structures. Then we give the basic definitions in combinatoric of words (cf. section 1.3) and in the theory of automata (cf. section 1.4). We conclude in section 1.4 with some standard definitions and properties on graphs, pointing out the breadth-first procedure for visiting a graph, that will be useful for what follows.

1.1 Monoids

A *semigroup* is a non trivial set S with a binary internal operation $\cdot : S \times S \longrightarrow S$ that is associative: for each $a, b, c \in S$ it is $(a \cdot b) \cdot c = a \cdot (b \cdot c)$. For simplicity of writing we will omit to indicate the operation $\cdot$ and we will use just the concatenation of the elements of S.

A *monoid* is a semigroup M equipped with a neutral element, that is an element $1_M \in M$ such that, for each $a \in M$, it is $a1_M = 1_M a = a$. The neutral element is unique and it is usually denoted by 1 or 1_M. A *submonoid* of M is a subset N which is stable under the operation and which contains the neutral element of M. So, $N \subseteq M$ is a submonoid of M if, for each $n, m \in N$, it is $nm \in N$ and if $1_M \in N$.

Definition 1.1.1 *Let M be a monoid. A subset of M, $\{m_i\}_{i \in I} \subseteq M$, is a set of generators for M if, for each $m \in M$, there exist $m_1, \ldots, m_k \in \{m_i\}_{i \in I}$ such that $m = m_1 m_2 \cdots m_k$.*

A monoid is *finitely generated* if it admits a finite set of generators. We say that M is the *trivial monoid* if it consists only of the neutral element, $M = \{1\}$. We say that

a monoid M is a *cyclic monoid* if it is generated by an element. All these definitions are extended to submonoids of a monoid M.

Let M and M' be two monoids and let $\varphi : M \longrightarrow M'$ be a map. We say that φ is a *morphism of monoids* if, for each $m_1, m_2 \in M$ it results $\varphi(m_1 m_2) = \varphi(m_1)\varphi(m_2)$ and if $\varphi(1_M) = 1_{M'}$. If φ is a bijection then we say that φ is an isomorphism.

We introduce now an important class of monoids: the free monoids. We will define it by means of a universal property. Let A be a finite non-empty set that we call *alphabet*.

Definition 1.1.2 *Let A be an alphabet and M a monoid. Then M is free on A if A is a subset of M and, for every mapping $\varphi : A \longrightarrow N$ into a monoid N, there exists a unique morphism of monoids $\varphi^* : M \longrightarrow N$ which extends φ.*

In this case we say that M is free and A is a basis of M.

It is equivalent to say that M is a free monoid with basis A if, and only if, every $m \in M$ can be written in a unique way in product of elements of A. It is proved that any free monoid admits only one basis. So, the cardinality of the basis A is called the *rank* of M and it is denoted by $rk(M)$.

The universal property of free monoids defines a unique free monoid on a given set A:

Proposition 1.1.3 *Let M and M' be free monoids on the alphabet A. Then M and M' are isomorphic.*

Given an alphabet A, we define now the monoid A^* of words on A.

Let A be an alphabet. The elements of A are called *letters*. A *word* w on the alphabet A is a finite sequence of elements of A

$$w = (a_1, a_2, \cdots, a_n), \quad a_i \in A$$

The set of all words on the alphabet A is denoted by A^* and is equipped with the associative operation defined by the concatenation of two sequences

$$(a_1, a_2, \cdots, a_n)(b_1, b_2, \ldots, b_m) = (a_1, a_2, \cdots, a_n, b_1, b_2, \ldots, b_m)$$

This operation is associative. This allow us to write

$$w = a_1 a_2 \ldots a_n$$

instead of $w = (a_1, a_2, \cdots, a_n)$. The empty sequence is called the *empty word* and is denoted by ε. It is the neutral element for concatenation. Thus the set A^* of words is equipped with the structure of a monoid. Since every word can be written as product of letters of A in a unique way, it follows that A^* is free with basis A. The monoid A^* is called *the free monoid on A*. This is a good denomination since every free monoid on A is isomorphic to A^*, so A^* is, in such a way, representative of the free monoids on

A. The set of non-empty words of A^* is denoted by A^+. The *length* (w) of the word $w = a_1 \dots a_n$ with $a_i \in A$ is the number n of letters of w. Clearly $(\varepsilon) = 0$.

Let us consider, from now on, the free monoid on a finite alphabet A. Every submonoid of A^* is not necessarily a free submonoid, as we can see in the following example.

Example 1.1.4 Let $A = \{a, b\}$ and let $X = \{ab, ba, a\}$. The submonoid X^* of A^* is not free on X because there exists $w = aba = a(ba) = (ab)a$.

On the contrary we will see that every subgroup of a free group is free. If M is a free submonoid on X of A^* then we have defined the basis of M as X. In all cases, to every submonoid M of A^* we can associate a unique minimal set of generators, that is a set of generators not containing properly another set of generators.

Proposition 1.1.5 *Let A be an alphabet. Any submonoid M of A^* has a unique minimal set of generators $X = (M - \varepsilon) - (M - \varepsilon)^2$.*

We can so define the *rank* of a submonoid M of A^* as the cardinality of the minimal set of generators, it is denoted by $rk(M)$. If M is free, then the basis of M is also the minimal set of generators. If M is trivial then $rk(M) = 0$. We also define the *reduced rank* of a submonoid M as $\widetilde{rk}(M) = max(rk(M) - 1, 0)$. So, if M is not trivial then $\widetilde{rk}(M) = rk(M) - 1$ and if M is trivial $\widetilde{rk}(M) = 0$.

Let us consider now the *intersection* of two submonoids of A^*. The intersection of two free submonoids is still a free submonoid. The intersection of two finitely generated submonoids it can be finitely generated or not.

Example 1.1.6 Let $A = \{a, b\}$. Let $H = \{aab, aba\}^*$ and $K = \{a, baaba\}^*$. One has $H \cap K = \{a(abaaba)^*baaba\}^*$ that is not finitely generated.

Let $H' = \{aba, bb\}^*$ and $K' = \{a, baaba\}^*$. One has $H \cap K = \{abaaba\}^*$ that is finitely generated.

1.2 Groups

A *group* is a monoid G such that every element admits an inverse, that is for each $g \in G$ there exists an element $\overline{g} \in G$ such that $g\overline{g} = 1_G = \overline{g}g$. The inverse of each element g is unique and it is usually denoted by g^{-1}.

A *subgroup* of G is a submonoid of G which contains the inverse of each element. Another way to say is that $H \subseteq G$ is a subgroup of G if, for each $h, g \in H$, it is $hg^{-1} \in H$.

Many definitions we have given for monoids can be given also for groups, as the notion of free group on an alphabet:

Definition 1.2.1 *Let A be an alphabet and F a group. Then F is free on A if A is a subset of F and, for every mapping $\varphi : A \longrightarrow G$ into a group G, there exists a unique morphism of groups $\varphi^* : F \longrightarrow G$ which extends φ.*

In this case we say that F is free and A is a basis of F.

Free groups exist in each alphabet, and are uniquely defined:

Proposition 1.2.2 *Let F and F' be free groups on the alphabet A. Then F and F' are isomorphic.*

Let us now define $F(A)$ *the free group on* A alphabet. Let $\overline{A} = \{\overline{a} \mid a \in A\}$ be a disjoint copy of A and let us extend the map $- : A \cup \overline{A} \longrightarrow A \cup \overline{A}$ to an involution of $A \cup \overline{A}$ by letting $\overline{\overline{a}} = a$, for each $a \in A$.

We say that the word u reduces to the word v by an elementary reduction, $u \xrightarrow{1} v$, if $u = xa\overline{a}y$ and $v = xy$, for some $x, y \in (A \cup \overline{A})^*$ and $a \in (A \cup \overline{A})$. If $k \geq 1$, we say that u reduces to v in $k+1$ steps, $u \xrightarrow{k+1} v$, if there exists a word w such that $u \xrightarrow{k} w$ and $w \xrightarrow{1} v$. We also say that u reduces to v in 0 steps if $u = v$. Finally, we say that u *reduces to* v, $u \longrightarrow v$, if u reduces to v in k steps, with $k \geq 0$.

A *reduced word* is a word that cannot be reduced anymore, that is a word which does not contain a factor of the form $a\overline{a}$, for each $a \in (A \cup \overline{A})$.

We define $F(A)$ as the set of reduced words over the alphabet A. It is proved that $F(A)$ is a free group with basis A.
We have that every word $u \in (A \cup \overline{A})^*$ reduces to a unique reduced word, we denote it by $red(u)$. We denote by $|u|$ the length of $red(u)$. We instead denote by (u) the length of the word u.

From here to the following, we will denote by A^{-1} the set $\overline{A}$ and, for each $a \in (A \cup \overline{A})$, by a^{-1} the element $\overline{a}$.

Let us recall now some properties of free groups. Differently from what happens in monoids, any subgroup of a free group is free. Moreover the intersection of two finitely generated subgroups is finitely generated.

In definition 1.2.1 we have defined a basis of a free group. Differently from what happens in the free monoid on A, where A is the unique basis, a free group on A can have different bases.

For instance A^{-1} is another basis of $F(A)$. Moreover, for each non-trivial automorphism φ of $F(A)$ (that is an isomorphism from $F(A)$ to $F(A)$), $\varphi(A)$ is a basis of $F(A)$. Anyway all bases of a free group F have the same cardinality so we can define the *rank* of a free group F as the cardinality of a basis of F, it is denoted by $rk(F)$. One can moreover define the *reduced rank* of a free group F as $\widetilde{rk}(F) = max(rk(F) - 1, 0)$. Since, every subgroup of a free group is free, then one can defined the rank and the reduced rank of a subgroup of a free group.

There exist interesting bases in $F(A)$ that are bases in which the *cancellation* in the products of words of such bases is reduced to the minimum. These bases are called Nielsen reduced bases and strongly Nielsen reduced bases. For definitions and properties of such bases we refer to [**?**].

Let $F(A)$ be a free group. Every element u of $F(A)$ is a reduced word in $(A \cup A^{-1})^*$. Every element in $F(A)$ can be seen as a word in $(A \cup A^{-1})^*$ or as an algebraic element in $F(A)$. The point of view will be understandable by the context.

In general, given $u, v \in F(A)$, there exist unique $u_1, v_1, u_2 \in F$ such that $u = u_1 u_2$, $v = {u_2}^{-1} v_1$ and the word $u_1 v_1$ is reduced. We say that the parts u_2 of u and ${u_2}^{-1}$ of v have been cancelled.

Proposition 1.2.3 *Let $u, v \in F(A)$. There exist unique $u_1, v_1, u_2 \in F(A)$ such that $u = u_1 u_2$, $v = {u_2}^{-1} v_1$ and the word $u_1 v_1$ is reduced.*

Let $U \subseteq F(A)$ be a subset of $F(A)$. We define the subset $U^{\pm}$ of $F(A)$ as $U^{\pm} = U \cup U^{-1}$. Let us define now the Nielsen reduced sets.

Definition 1.2.4 *$U \subseteq F(A)$ is a Nielsen reduced set if:*

1)for each $u \in U$, $u^{-1} \notin U$

2)for each $u, v, w \in U^{\pm}$, $uv \neq 1$ and $vw \neq 1$, it is $|uvw| > |u| - |v| + |w|$

In particular, a Nielsen reduced set is a set in which in every product uvw, with $uv \neq 1$ and $vw \neq 1$, not all of v is cancelled. Let us define now the strongly Nielsen reduced sets. These are, in particular, Nielsen reduced sets with a condition on the cancellation in the product of two words.

Definition 1.2.5 *$U \subseteq F(A)$ is a strongly Nielsen reduced set if:*

1)for each $u \in U$, $u^{-1} \notin U$

2)for each $u, v, w \in U^{\pm}$, $uv \neq 1$ and $vw \neq 1$, it is $|uvw| > |u| - |v| + |w|$

3)for each $u, v \in U^{\pm}$, $uv \neq 1$, it is $|uv| \geq max(|u|, |v|)$

In particular, a strongly Nielsen set has the property that in every product uv it is not cancelled more than the half of u and of v. We will study such bases more in detail in chapter 3.

In the past years it was posed the problem of finding, given a finite set of elements U in a free group F, an equivalent set (i.e. a set generating the same subgroup as U) that was a strongly Nielsen set. Such a problem is called *the strongly Nielsen reduction*. The algorithms known in literature work on words (see [14], [1]). In particular they utilize some transformations on words: the Nielsen transformations.

In particular, given U a finite subset of F, free group, we think of it as an ordered set: $U = \{u_1, \ldots, u_n\}$. We denote by U^{-1} the set of all the inverses of the elements in U, and by $U^{\pm} = U \cup U^{-1}$.

The *three elementary Nielsen transformations* on $U = \{u_1, \ldots, u_n\}$ are defined as follows:

(T1) replace some u_i by u_i^{-1};

(T2) replace some u_i by $u_i u_j$ where $j \neq i$;

(T3) delete some u_i where $u_i = 1$.

A product of such transformations is a *Nielsen transformation*. One has that if U is carried into V by a Nielsen transformation then U and V are equivalent, that is they generate the same subgroup.

In [14] it has been proved the following proposition:

Proposition 1.2.6 *If $U = \{u_1, \ldots, u_n\}$ is a finite subset of F free group, then U can be carried by a Nielsen transformation into some V such that V is strongly Nielsen reduced.*

The algorithm for the strongly Nielsen reduction with the best complexity was found by Avenhaus and Madlener in 1984 (cf. [1]). In particular, they find an algorithm that, given $U = \{u_1, \ldots, u_m\} \subseteq F(A)$ with $|u_i| \leq n$, constructs an equivalent strongly Nielsen set in time $O(m^5 n^2)$.

1.3 Words and Codes

This section contains the definitions of codes, prefix and suffix codes and the correspondence between codes and free submonoids.

We say that a word $u \in A^*$ is a *factor* of the word $w \in A^*$ if there exist $v_1, v_2 \in A^*$ such that $w = v_1 u v_2$. The factor u is *proper* if $u \neq w$. We say that a word $u \in A^*$ is a *prefix* (resp. *suffix*) of the word $w \in A^*$ if there exists $v \in A^*$ such that $w = uv$ ($w = vu$). The prefix (resp. suffix) u is proper if $u \neq w$. Given two words $u, v \in A^*$ we say that w is the *greatest prefix in common between u and v* if w is a prefix of u and a prefix of v and it does not exist a prefix of u and v greater than w. Analogously one can define the *greatest suffix in common between u and v*.

Let A be an alphabet. A subset X of the free monoid A^* is a *code* if every word in X^+ can be written in a unique way as a product of words in X.

An important class of codes is the class of prefix codes. Let X be a subset of A^*. Then X is a *prefix set* (resp. *suffix set*) if no element of X is a proper prefix (resp. suffix) of another element in X. A set X is a *bifix set* if it is both prefix and suffix.

Proposition 1.3.1 *Any prefix (suffix, bifix) set of words $X \neq 1$ is a code.*

The fact that X is a code is equivalent to the property that X^* is a free monoid. We have already given the proposition that states the uniqueness of the minimal set of generators for a submonoid H of A^*. If H is free it follows that the minimal set of generators of H is a code and conversely if X is a code then X^* is a free submonoid with X as minimal set of generators. The following proposition resumes this equivalence between codes and free submonoids.

Proposition 1.3.2 *If M is a free submonoid of A^*, then its minimal set of generators is a code. Conversely, if $X \subseteq A^*$ is a code, then the submonoid X^* of A^* is free and X is its minimal set of generators.*

In particular, if X is a prefix set then X^* is a free submonoid on X.

1.4 Automata and Graphs

1.4.1 Automata: basic definitions

In this subsection we recall the basic definition of automaton. For the notation we refer to [2].

Let A be an alphabet. We consider finite states automata on an alphabet A.
An *automaton* over A,

$$\mathcal{A} = (Q, I, T, \mathcal{F})$$

consists of a finite set Q of *states*, of two subsets I and T of Q called sets of *initial* and *final* states, respectively, and of a set

$$\mathcal{F} \subset Q \times A \times Q$$

whose elements are called *edges*.
An edge

$$e = (x, a, y)$$

is also denoted by

$$e : x \xrightarrow{a} y$$

The letter a is called the *label* of the edge. We will say that either the edge e goes out from x and comes in y or e start at x and end at y.

Two edges $e : x \xrightarrow{a} y$ and $g : x' \xrightarrow{b} y'$ are *consecutive* if $y = x'$. A *path* in $\mathcal{A}$ is a finite sequence

$$p = p_1 p_2 \dots p_n$$

of consecutive edges $p_i : x_i \xrightarrow{a_i} x_{i+1}$. We shall also write

$$p : x_1 \xrightarrow{w} x_{n+1}$$

where

$$w = a_1 a_2 \dots a_n$$

is the *label* of the path p. The path p is said to start at x_1 and end at x_{n+1}. We indicate by $i(p) = x_1$ the starting state and by $f(p) = x_{n+1}$ the ending state. The *length* of a path is the number of edges that compose it. For each state $x \in Q$ it is defined the *null path* starting and ending at x, denoted by $1_x : x \longrightarrow x$. Its length is 0 and its label is $\varepsilon \in A^*$.

When two paths p and q are consecutive (i.e. $f(p) = i(q)$) then p and q can be concatenated and we call the resulting path pq. A *subpath* of a path p is a subsequence of consecutive edges. A subpath of a path p is a *prefix* of p if it starts at the same starting state of p. Given two paths p and q starting at the same state x, the *longest prefix path in common* between p and q is a path prefix of p and prefix of q that is the longest with this property. Analogously it can be defined, given two paths p and q ending in the same state x, the *longest suffix path in common* between p and q.

A path p is a *simple path* if all states in the path are distinct. Given two states x and y, if there exists a path from x to y then there exists also a simple path from x to y. A path p is a *cycle* if it is not the null path and if it starts and ends at the same state. We say that a path c is a cycle in x if it starts and ends at x. We moreover say that c is a cycle with basis x. A cycle c is a *simple cycle* if it has all the intermediate states distinct and different from the starting state.

We now introduce a new nomenclature for the cycles that do not visit the starting state in the intermediate states:

Definition 1.4.1 *Given a cycle c, we say that c is* simple in x *if it is a cycle in x such that no intermediate state is equal to x.*

We will see that this nomenclature will be useful for the following chapter.

Remark 1.4.2 *We remark that a cycle that is simple in x is not in general a simple cycle.*

Example 1.4.3 *In the automaton in figure 1.1 the cycle $c : 1 \xrightarrow{a} 2 \xrightarrow{a} 2 \xrightarrow{b} 1$ is a cycle simple in 1 but it is not a simple cycle. The cycle $c' : 1 \xrightarrow{a} 2 \xrightarrow{b} 3 \xrightarrow{a} 1$ is a simple cycle so, in particular, it is a cycle simple in 1.*

Let $\mathcal{A} = (Q, I, T, \mathcal{F})$ be an automaton. The *language recognized by* $\mathcal{A}$, $L(\mathcal{A})$, is the set of words that are labels of paths from an initial state to a final one. A state $x \in Q$ is *accessible* (resp. *coaccessible*) if there is a path $p : i \longrightarrow x$ with $i \in I$ (resp. a path $p : x \longrightarrow t$ with $t \in T$). An automaton $\mathcal{A}$ is a *trim* automaton if all the states of the automaton are accessible and coaccessible.

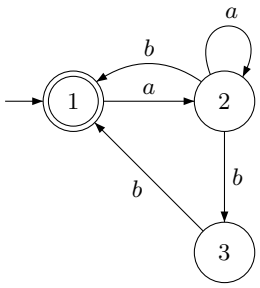

Figure 1.1: $\mathcal{A}$ automaton

An automaton $\mathcal{A} = (Q, I, T, \mathcal{F})$ is *unambiguous* if, for each $x, y \in Q$, for each $w \in A^*$, there exists at most one path starting at x and ending at y with label w.
An automaton $\mathcal{A}$ over A is a *deterministic automaton* if $card(I) = 1$ and if

$$(x, a, y), (x, a, y') \in \mathcal{F} \implies y = y'$$

that is, for each state x and for each $a \in A$, there is at most one edge starting in x with label a. We will denote by i such unique initial state. Given $\mathcal{A} = (Q, i, T, \mathcal{F})$, a deterministic automaton, it can be defined the function

$$\delta : Q \times A \longrightarrow Q$$

$$\delta(x, a) = \begin{cases} y & \text{if} \quad (x, a, y) \in \mathcal{F}; \\ \emptyset & \text{otherwise.} \end{cases}$$

This function δ is extended to words in A^* by setting, for all $x \in Q$, $\delta(x, \varepsilon) = x$ and, for $w \in A^*$ and $a \in A$, $\delta(x, wa) = \delta(\delta(x, w), a)$. This function is called the *transition function*. With this notation we have

$$L(\mathcal{A}) = \{w \in A^* \mid \delta(i, w) \in T\}$$

Let $\mathcal{A} = (Q, I, T, \mathcal{F})$ be an automaton. We say that a state $x \in Q$ is a *branch point going out* (in short *bpo*) if the number of edges going out from x is at least two. We say that $x \in Q$ is a *branch point going in* (in short *bpi*) if the number of edges coming in x is at least two. In the automaton in figure 1.1 the state 2 is a bpo and the states 1 and 2 are bpi's.

Remark 1.4.4 *From now on we will consider automata $\mathcal{A}$ with non empty languages and different from the trivial language, that is $L(\mathcal{A}) \neq \emptyset$ and $L(\mathcal{A}) \neq \{\varepsilon\}$.*

1.4.2 Graphs, trees: definition and properties

In this subsection we recall some definitions on graphs and multigraphs and the algorithm of the breadth-first search, procedure for visiting a graph. For definitions and notations we refer to [9].

A *directed graph* $\Gamma = (V, E)$ is a pair (V, E) where V is a finite set and $E \subseteq V \times V$ is a binary relation on V. A *directed multigraph* is like a graph but it can have more than one edge between two vertices. More formally a directed multigraph is a 4-uple (V, E, i, f) where V and E are two disjoint sets and $i, f : E \longrightarrow V$ are maps. In particular, for each $e \in E$, $i(e)$ denotes the initial vertex of e and $f(e)$ denotes the final vertex of e.

A *directed labelled graph* $\Gamma = (V, E)$ on A alphabet is a pair (V, E) where V is a finite set and $E \subseteq V \times A \times V$.

An *undirected graph* is defined like a directed graph but the edges are sets of two vertices. In an undirected graph self-loops (edges from a vertex to itself) are not admitted. An *undirected multigraph* is like a graph but it can have more than one edge between two vertices and self-loops. More formally an undirected multigraph is a 4-uple (V, E, f) where V and E are two disjoint sets and $f : E \longrightarrow \{\{u, v\} \mid u, v \in V\}$ are maps. In particular, for each $e \in E$, $f(e)$ denotes the set of extremal vertices of e. We have already given the definitions of paths and cycles for automata that are in particular directed labelled graphs. Some of such definitions are a little bit different in undirected graphs and we recall here such basic definitions.

Let $\Gamma = (V, E)$ be an undirected graph. A *path* of length n is a sequence of vertices $(v_0, \ldots, v_n)$ such that, for each i, $\{v_{i-1}, v_i\} \in E$. Since our paths are undirected throughout we shall identify the paths $p = (v_0, \ldots, v_n)$ and $p^{-1} = (v_n, \ldots, v_0)$. Since a path in Γ does not contain self-loops then $v_i \neq v_{i+1}$ for each i.

Given a path $(v_0, \ldots, v_n)$, we say that, for each $i = 1, \ldots, n$, the path p contains the edge $e_i = \{v_{i-1}, v_i\}$, and that the edges $e_1, \ldots, e_n$ occur in p. Given $p = (v_0, \ldots, v_n)$ and $q = (u_0, \ldots, u_m)$ with $v_n = u_0$, it is defined the path pq concatenation of p and q as $pq = (v_0, \ldots, v_n, u_1, \ldots, u_m)$.

A path is *simple* if all vertices in the path are distinct. A *cycle* in Γ is a path of length at least 3 with the same initial and final vertices such that all the intermediate vertices are distinct. We say that *a cycle visits a vertex* x if x is one of its intermediate vertices. An undirected graph without cycles is said to be *acyclic*. Given a directed or undirected graph $\Gamma = (V, E)$ and given $x, z \in V$, we say that x is *reachable* from z if there exists a path in Γ from z to x. The graph $\Gamma = (V, E)$ is *connected* if, for each $x, y \in V$, there exists a path from x to y.

Example 1.4.5 *Considering the graph in figure 1.2 as undirected, the path given by the sequence of vertices* $(2, 3, 4, 2)$ *is a cycle while the path given by the sequence of vertices* $(1, 2, 3, 4)$ *is not a cycle.*

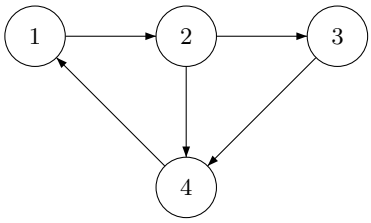

Figure 1.2:

A *tree* is an undirected graph which is connected and acyclic. Given $T = (V(T), E(T))$ subgraph of Γ, we say that *a path p of Γ is in T* if it contains only edges of $E(T)$. Given a graph $\Gamma = (V, E)$, a *spanning tree* of Γ, $T = (V(T), E(T))$ is a subgraph of Γ which is a tree such that $V(T) = V$.

Proposition 1.4.6 *Let $T = (V, E)$ be a tree. Let p be a path in T such that for each $e \in E$, ee does not occur in p then p is simple.*

Proof. Let T be a tree. Let $p = (v_0, \ldots, v_n)$ be a path in T such that for each $e \in E$, ee does not occur in p.

If p is not simple then there exist v_i, v_j vertices in p such that $v_i = v_j$ and, for each $i \leq k, l \leq j$, $(i, j) \neq (k, l)$, it is $v_k \neq v_l$. Let $p' = (v_i, \ldots, v_j)$ be the sub-path of p starting at v_i and ending at v_j. There exist at least two vertices in p' different from v_i since $v_i \neq v_{i+1}$ and ee is not contained in p, for each $e \in E$. So p' is a path with length at least 3 and the same initial and final vertex and all the intermediate vertices distinct, and so it is a cycle in T. This is a contradiction since T is acyclic. □

Example 1.4.7 *Considering the tree in figure 1.3 as undirected, the path given by the sequence of vertices $(1, 2, 3)$ does not contain any edge twice consecutively and it is a simple path. The path $(1, 2, 1, 2, 3)$ it is not a simple path and it contains the edge $(1, 2)$ twice consecutively.*

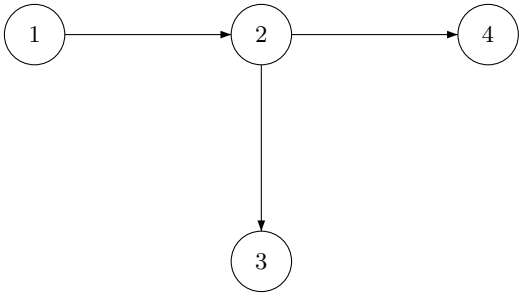

Figure 1.3:

Remark 1.4.8 *In a simple path every edge occurs just once.*

There exist different algorithms for searching a graph either direct of undirect. One of the simplest ones and most known is the *breadth-first search* (BFS in shorts). Given a directed or undirected graph $\Gamma = (V, E)$ and a distinguished *source* vertex s, BFS systematically explores the edges of Γ to "discover" every vertex that is reachable from s. It also produces a tree T called *breadth-first tree* with root s that contains all such reachable vertices. Moreover, for each x reachable from s, there is a unique simple path from s to x in T, that is also one of the "shortest paths" from s to x in Γ.

Breath-first search is so called because it expands the frontier between discovered and undiscovered vertices uniformly across the breadth of the frontier.

The BFS procedure assumes that the input graph is represented using adjacency lists. The *adjacency-list representation* of a graph $\Gamma = (V, E)$ consists of an array Adj of $|V|$ lists, one for each vertex in V. For each $x \in V$, the adjacency-list $Adj[x]$ contains pointers to all the vertices y such that there is one edge $(x, y) \in E$.

Let us give now a brief and informal description of the algorithm. To keep track of the progress, BFS colors each vertex white, gray and black. At the beginning all the vertices are colored white. A vertex is *discovered* the first time it is encountered during the search, at which time it becomes gray or black. So, gray and black vertices are already discovered, but BFS distinguish between them to make the search proceeding in a breadth-first way. If $(x, y) \in E$ and x is black, then y is gray or black, that is y has been discovered. Instead gray vertices may have some adjacent white vertices. Gray vertices represent the frontier between discovered and undiscovered vertices.

Remark 1.4.9 *The BFS procedure can be done also for either directed or undirected multigraphs.*

Remark 1.4.10 *The BFS procedure can be done also for automata.*

In fact an automaton is in particular a directed multigraph and BFS procedure can be done also for directed multigraphs.

As anticipated, breadth-first search constructs a breadth-first tree, initially containing only its root, which is the source vertex s. Whenever a white vertex x is discovered, while scanning the adjacency list of a vertex y, then the vertex x and the edge (x, y) are added to the breadth-first tree.

The BFS procedure discovers every vertex just once, so in $T = (V(T), E(T))$ there is only one edge ending at a fixed vertex and so $E(T) = V(T) - 1$. Since T is connected, this implies that T is a tree. We are in particular interested in the properties of this tree that are resumed in the following proposition:

Proposition 1.4.11 *Let $\Gamma = (V, E)$ be a graph. Let $s \in V$. The breadth-first tree $T = (V', E')$ of Γ in s, obtained applying the BFS to Γ in s, is such that:*

1) T is a tree

2) V' consists of all vertices reachable from s

3) For all $v \in V'$ there is a unique simple path from s to v in T, that is also one of the shortest paths from s to v in Γ.

It is well known (cf. [9]) that, if $\Gamma = (V, E)$ and $s \in V$, then the total running time of the breadth-first search of Γ in s, and so of the construction of the BFS tree, is $O(|V| + |E|)$.

Remark 1.4.12 *We remark that the tree obtained applying the BFS procedure to a multigraph Γ with V vertices and E edges constructs a BFS tree satisfying the properties in prop. 1.4.11 in total running time $O(|V| + |E|)$.*

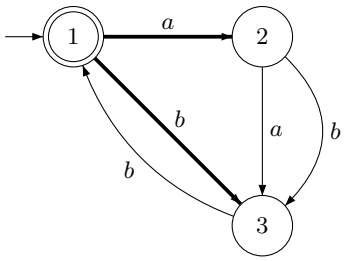

Figure 1.4: $\mathcal{A}$ monoidal automaton. T BFS tree in 1

Example 1.4.13 *In figure 1.4 we see an example of a monoidal automaton with pointed out the edges of the breadth-first tree in* 1.

Chapter 2

Automata and submonoids

In this chapter we consider a correspondence between submonoids of the free monoid on A and monoidal automata on A. Such automata on A are trim automata with a unique initial-final state recognizing submonoids of A^*. We will deepen the study of such correspondence examining some classes of such automata.

In particular, defining a *semi-flower automaton* as a monoidal automaton in which all cycles visit the initial-final vertex, we get that semi-flower automata correspond to finitely generated submonoids.

Our purpose is to study the intersection of two finitely generated submonoids, by studying the product of two monoidal automata recognizing such submonoids. This study is more difficult with respect to the case of free groups, since the intersection of two finitely generated submonoids can be infinitely generated and, in the case of finite rank, it is not so easy to count the number of generators of such intersection.

This last problem gives rise to a more general problem, that is:

-Given a semi-flower automaton $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ recognizing a submonoid H, how to give a general formula for the rank of H depending on the characteristics of the automaton?

We have solved the problem in the case of one bpi of $\mathcal{A}$. In this case we have that

$$rk(H) \leq |\mathcal{F}| - |Q| + 1$$

This inequality is strictly related of the one already known in free groups. In particular, if $\mathcal{A}$ is an 'inverse automaton' on A, that is a special automaton recognizing a subgroup $\overline{H}$ of $F(A)$, and $\Gamma_{\mathcal{A}} = (Q, \mathcal{F})$ is a 'special' graph associated to $\mathcal{A}$, we have that $rk(\overline{H}) = |\mathcal{F}| - |Q| + 1$.

Using such inequality and following the same arguments used in free groups, we get very interesting results in the case of prefix sets of generators and of product automata of a certain 'topology' (in particular with a unique bpi). In this case we get an upper

bound for the intersection's rank, that is:

$$\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$$

When H and K satisfy this inequality we say that H and K satisfy the *Hanna Neumann property*. There exist in general H and K with finite intersection's rank such that this property is not true and we bring a class of examples in section 2.3.3. In the case of more bpi's in the product automaton we discuss several approaches. As an application of the obtained results we find the Karumäki's characterization of the intersection of two submonoids generated by prefix sets of two words.

In section 2.1.1, we recall the definition of some standard automata associated to submonoids of A^*: the flower automaton and the literal automaton. In section 2.1.2, we study the monoidal automata pointing out an important subclass of such automata that is the subclass of semi-flower automata. These are a generalization of the flower automata and characterize the finitely generated submonoids.

In this way we build a correspondence between monoidal automata on A and submonoids of A^*. In this correspondence to semi-flower automata it is associated the class of finitely generated submonoids. Moreover, submonoids of A^* generated by finite prefix sets correspond to deterministic semi-flower automata.

In section 2.2 we study semi-flower automata with at most one bpi. In this case we obtain a link between the rank of the submonoids recognized by such automata and the characteristics of the automata. This link will allow us to prove in section 2.3 that submonoids generated by finite prefix sets such that the product automaton is semi-flower with a unique bpi satisfy the Hanna Neumann property. If the product automaton is a semi-flower automaton with more than one bpi then there exist H and K that do not satisfy the Hanna Neumann property (see section 2.3.3).

In section 2.5, we conclude with the proof of the result of Karhumäki for submonoids generated by prefix sets of two words, that is:

Theorem 2.0.14 *Let H and K be submonoids generated by prefix sets of two elements.*

1. *If $H \cap K$ is finitely generated then $rk(H \cap K)$ is at most two.*
2. *If $H \cap K$ is not finitely generated then there exist $u, v, w \in A^*$ such that $H \cap K = (u(v)^*w)^*$.*

In the general case, the idea is to research an upper bound, near to that one of the Hanna Neumann property, for product automata with more than one bpi, searching a link for semi-flower automata with more than one bpi between the rank and the characteristics of such automata. We conclude, in section 2.5, with a discussion on the non-prefix case.

2.1 The submonoids-automata correspondence

In this section we define the correspondence between submonoids and monoidal automata. We moreover analyze some closed classes in this correspondence.

We begin, in the following subsection, by recalling the definitions of several automata associated to finitely generated submonoids of A^*. In the subsection 2.1.2 we will made explicit such correspondence.

2.1.1 Automata recognizing submonoids

Let A be an alphabet and let X^* be a finitely generated submonoid of A^*. There exist several automata recognizing X^*. The most known (cf. [2], [7]) is the *flower automaton of* X. If X is a finite prefix set there is, moreover, another well known automaton recognizing X^* that is the *literal automaton of* X^*. In the following we give the definition of such automata.

To each submonoid X^* of A^* generated by a finite set X it is associated $\mathcal{F}_X$ the *flower automaton of* X (cf. [2],[7]).

It is built in the following way. First we build $\mathcal{S}_X$, the *solar automaton of* X recognizing X, in this way: we build one automaton for each word $x \in X$ with $(x)+1$ states (where (x) is the length of x) and merge all the initial states. Note that this automaton is a tree with root the initial state 1. Then we merge all the final states with the initial state 1. Doing this we obtain the flower automaton of X. Such an automaton is a trim automaton with a unique final state equal to a unique initial one recognizing X^*. Moreover it is such that all the cycles visit the unique initial-final state 1, all the cycles that are simple in 1 intersect themselves only in 1 and have as labels the words of X. See an example in figure 2.1.

Let now X be a finite prefix set of A^*. The *literal automaton of* X^* (cf. [2]) is the automaton

$$\mathcal{A}_X = (Q, \varepsilon, \varepsilon, \delta)$$

where $Q = X(A^+)^{-1}$ is the set of the proper prefixes of X and where

$$\delta(u,a) = \begin{cases} ua & \text{if } ua \in X(A^+)^{-1} \\ \varepsilon & \text{if } ua \in X \\ \emptyset & \text{otherwise} \end{cases}$$

It is immediate that $L(\mathcal{A}_X) = X^*$. See an example of literal automaton in figure 2.2.

Remark 2.1.1 *Let us note that in all figures every edge with label a, b has to be understood as two edges with labels a and b, respectively.*

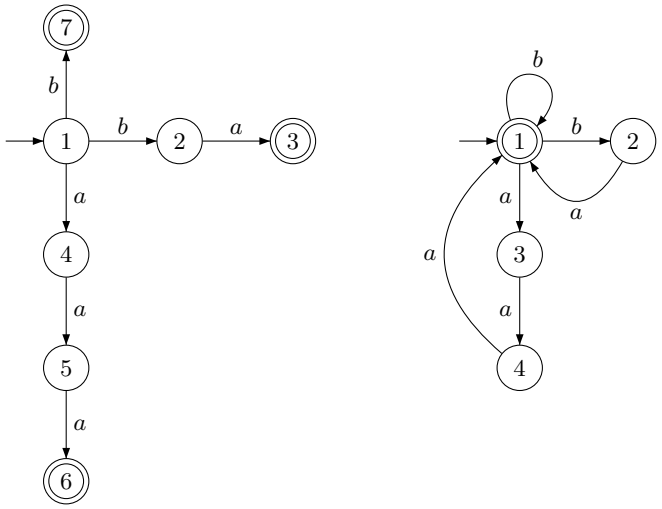

Figure 2.1: $\mathcal{S}_X$ the solar automaton of $X = \{b, ba, aaa\}$ and $\mathcal{F}_X$ the flower automaton of X

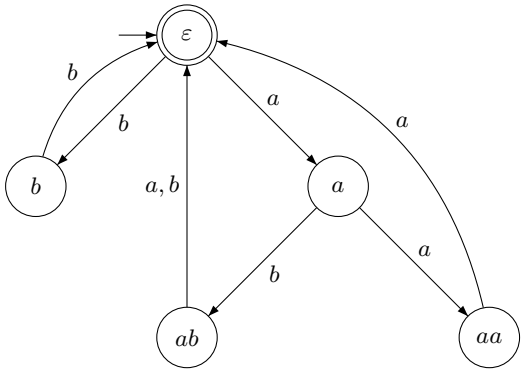

Figure 2.2: $\mathcal{A}_X$ literal automaton of X^*, where $X = \{bb, aba, abb, aaa\}$

2.1.2 Correspondence submonoids-monoidal automata

Let A be an alphabet. We define now a special kind of automata on A, the 'monoidal automata'. We will see, in the following, that these automata recognize submonoids of A^*.

Definition 2.1.2 *Let $\mathcal{A}$ be an automaton on A. We say that $\mathcal{A}$ is a* monoidal automaton *if it is a trim automaton with a unique final state equal to a unique initial one.*

If $\mathcal{A}$ is a monoidal automaton, we will denote by 1 the initial-final state of $\mathcal{A}$. See an example of monoidal automaton in figure 2.3. Let us note that in a monoidal

automaton, for each state x, there exist a simple path from 1 to x and a simple path from x to 1.

It is easy to prove that a monoidal automaton on A recognizes a submonoid of A^*, as stated in the following (cf. [2]):

Proposition 2.1.3 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton. Then $\mathcal{A}$ recognizes the submonoid generated by the set of labels of the cycles that are simple in* 1.

So, the generators of a monoidal automaton are the labels of the cycles that are simple in 1. It follows that a monoidal automaton can recognize an infinitely generated submonoid, as we can see in the example in figure 2.4.

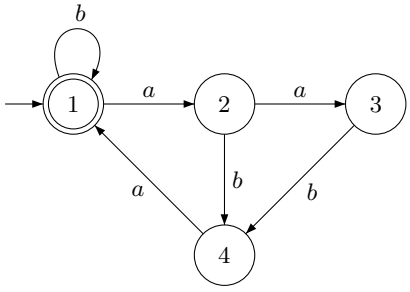

Figure 2.3: $\mathcal{A}$ monoidal automaton with $L(\mathcal{A}) = \{b, aba, aaba\}^*$

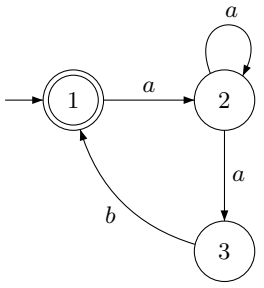

Figure 2.4: $\mathcal{A}$ monoidal automaton recognizing a submonoids infinitely generated by $Y_{\mathcal{A}} = \{aaa^*b\}$. One has $L(\mathcal{A}) = {Y_{\mathcal{A}}}^*$

Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton. Let us denote by $C_{\mathcal{A}}$ the set of cycles that are simple in 1 and by $Y_{\mathcal{A}}$ the set of their labels. In the example of figure 2.4 $Y_{\mathcal{A}} = \{aaa^*b\}$. As stated in prop. 2.1.3, it results $L(\mathcal{A}) = {Y_{\mathcal{A}}}^*$

In general, if $\mathcal{A}$ is monoidal then $Y_{\mathcal{A}}$ is not the minimal set of generators (see an example in figure 2.5). But, if we suppose that $\mathcal{A}$ is unambiguous then $Y_{\mathcal{A}}$ is the minimal set of generators and moreover the submonoid recognized by $\mathcal{A}$ is free.

Proposition 2.1.4 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be an unambiguous monoidal automaton. The automaton $\mathcal{A}$ recognizes a free submonoid with basis $Y_{\mathcal{A}}$.*

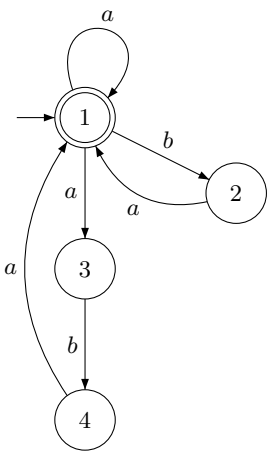

Figure 2.5: $\mathcal{A}$ monoidal automaton with $Y_{\mathcal{A}} = \{a, ba, aba\}$ not minimal set of generators since $L(\mathcal{A}) = \{a, ba\}^*$.

Proof. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be an unambiguous monoidal automaton. By prop. 2.1.3, $Y_{\mathcal{A}}$, the set of labels of cycles that are simple in 1 is a set of generators for $L(\mathcal{A})$. Moreover, by the unambiguity of $\mathcal{A}$, every element in $L(\mathcal{A})$ can be factorized in a unique way in words of $Y_{\mathcal{A}}$, otherwise there would be a word in $L(\mathcal{A})$ label of two different cycles in 1. □

The converse is also true:

Proposition 2.1.5 *Let $\mathcal{A}$ be a monoidal automaton on A recognizing a free submonoid of A^* with basis $Y_{\mathcal{A}}$. Then $\mathcal{A}$ is unambiguous.*

Proof. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton recognizing H, a free submonoid of A^* with basis $Y_{\mathcal{A}}$. If $\mathcal{A}$ is ambiguous then there exist two different paths with the same label from a state x to a state y. Since $\mathcal{A}$ is trim and with a unique initial-final state, considering two simple paths from 1 to x and from y to 1, we obtain two different cycles with the same label. This is not possible since H is free. □

To a monoidal automaton $\mathcal{A}$ on A it is associated the submonoid $H = L(\mathcal{A})$ of A^*. Conversely to each submonoid X^* of A^* ,generated by a finite set X, it is associated $\mathcal{F}_X$ the *flower automaton of* X (cf. [2], [7]). We have seen its definition in the previous subsection 2.1.1.

As we have seen, such an automaton is a monoidal automaton recognizing X^* such that all the cycles visit the unique initial-final state 1, all the cycles that are simple in 1 intersect themselves only in 1 and have as labels the words of X. In figure 2.5 $\mathcal{A}$ is the flower automaton associated to $X = \{a, ba, aba\}$.

Let us consider now finitely generated submonoids. The monoidal automata recognizing such submonoids have a special shape that recall that one of flower automata. Let us define now the semi-flower automata.

Definition 2.1.6 *Let $\mathcal{A}$ be an automaton. Then $\mathcal{A}$ is a semi-flower automaton if it is a monoidal automaton such that all the cycles visit the unique initial-final state.*

Hence in a semi-flower automaton the cycles that are simple in 1 intersect themselves not necessarily only in 1. It follows that the flower automaton associated to X, finite set of A^*, is a semi-flower automaton in which all the cycles that are simple in 1 intersect themselves only in 1.

Remark 2.1.7 *It is interesting to observe that in a semi-flower automaton every cycle that is simple in* 1 *is in particular a simple cycle.*

The following proposition states that semi-flower automata recognize finitely generated submonoids:

Proposition 2.1.8 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a semi-flower automaton, then $\mathcal{A}$ recognizes a finitely generated submonoid.*

Proof. Let $\mathcal{A}$ be a semi-flower automaton then, by prop. 2.1.3, $Y_{\mathcal{A}}$, the set of labels of the cycles that are simple in 1 is a set of generators for $L(\mathcal{A})$. By remark 2.1.7, such cycles are simple. Finally, the number of such cycles is finite since the automaton has a finite number of states. □

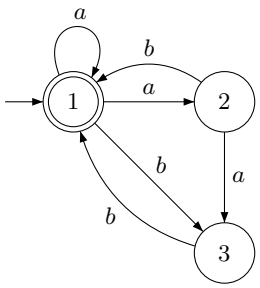

Figure 2.6: $\mathcal{A}$ semi-flower automaton. It recognizes a finitely generated submonoid: $L(\mathcal{A}) = \{a, ab, aab, bb\}^*$

An example of semi-flower automaton is shown in figure 2.6.
The converse of Proposition 2.1.8 is not true in general as it is shown in the example of figure 2.7. However, with the supplementary hypothesis of unambiguity we get also the converse as stated in the following proposition:

Proposition 2.1.9 *Let $\mathcal{A}$ be an unambiguous monoidal automaton such that $L(\mathcal{A}) = H$. The submonoid H is finitely generated if, and only if $\mathcal{A}$ is a semi-flower automaton.*

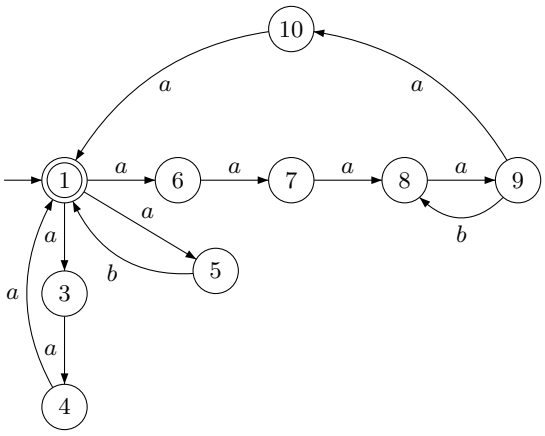

Figure 2.7: $\mathcal{A}$ ambiguous not semi-flower automaton that recognizes a finitely generated submonoid: $L(\mathcal{A}) = \{ab, aaa\}^*$

Proof. Let $\mathcal{A}$ be an unambiguous monoidal automaton that recognizes H finitely generated submonoid. By prop.2.1.4, H is free with basis $Y_{\mathcal{A}}$, the set of labels of the cycles that are simple in 1. By hypothesis,

$$|Y_{\mathcal{A}}| = rk(H) < \infty$$

If there exists a cycle c not visiting 1 then, since $\mathcal{A}$ is monoidal, there exists an infinite number of different cycles that are simple in 1, that is a contradiction.
The other implication is proved in prop. 2.1.8. □

Let us consider now deterministic monoidal automata. The submonoids recognized by these automata are generated by prefix sets.

Proposition 2.1.10 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a deterministic monoidal automaton. Then $\mathcal{A}$ recognizes a free submonoid generated by a prefix set.*

Proof. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a deterministic monoidal automaton. By prop.2.1.4 and prop.2.1.8 , $L(\mathcal{A})$ is a free submonoid with basis $Y_{\mathcal{A}}$. Let us prove that this is a prefix set. By contradiction, let u, v be labels of c_u and c_v, cycles that are simple in 1, such that $v = uw$ with w not empty word. $\mathcal{A}$ is deterministic so $c_v = c_u c_w$ with c_w the cycle with label w. This is a contradiction because c_v is a cycle simple in 1. □

For semi-flower automata the previous proposition becomes:

Corollary 2.1.11 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a deterministic semi-flower automaton. $\mathcal{A}$ recognizes a free submonoid generated by a finite prefix set.*

As seen in the previous subsection 2.1.1, given a submonoid X^* of A^*, generated by a finite prefix set X, we can construct an automaton recognizing X^*, that is $\mathcal{A}_X$, the literal automaton of X^*.

We prove now that the literal automata are deterministic semi-flower automata:

Proposition 2.1.12 *Let X be a finite prefix set then $\mathcal{A}_X$ is a deterministic semi-flower automaton.*

Proof. Let $\mathcal{A}_X = (Q, \varepsilon, \varepsilon, \delta)$. By construction $\mathcal{A}_X$ is a deterministic monoidal automaton. Since $L(\mathcal{A}_X) = X^*$, X is finite and $\mathcal{A}$ is deterministic then, by prop. 2.1.9, $\mathcal{A}_X$ is a semi-flower automaton. □

We have so found that if $\mathcal{A}$ is a deterministic semi-flower automaton, then $L(\mathcal{A})$ is a submonoid generated by a finite prefix set. Conversely, given a finite prefix set X, there exists a deterministic semi-flower automaton recognizing X^*, that is the literal automaton of X^*. Hence deterministic semi-flower automata recognize submonoids generated by finite prefix sets.

Let us now resume the obtained results in such correspondence.

Let $\mathcal{A}$ be a monoidal automaton recognizing a submonoid H, then:

-$\mathcal{A}$ is an unambiguos automaton if and only if H is a free submonoid with basis $Y_{\mathcal{A}}$

-If $\mathcal{A}$ is deterministic then H is a free submonoid generated by a prefix set

Let $\mathcal{A}$ be an unambiguous monoidal automaton recognizing a submonoid H, then:

-$\mathcal{A}$ is a semiflower automaton if and only if H is a finitely generated submonoid

2.2 Studying of automata with a certain number of bpi's

In this section we consider the sets of bpo's and bpi's of a monoidal automaton and we study semi-flower automata with a certain number of bpi's.

2.2.1 Definition and properties about branch points of an automaton

At the end of section 1.4 we have defined a branch point going out, in short bpo, (resp. branch point going in, in short bpi) of an automaton as a state x of the automaton where the number of edges going out from x is at least two (resp. the number of edges coming in x is at least two).

In this subsection we define some subsets of bpo's and bpi's and we give some formulas useful for what follows.

Let $\mathcal{A}$ be an automaton. Let $BPI(\mathcal{A})$ be the set of states of $\mathcal{A}$ that are bpi's and let $BPO(\mathcal{A})$ be the set of states of $\mathcal{A}$ that are bpo's.

Example 2.2.1 *The automaton in figure 2.8 is a monoidal automaton such that* $BPI(\mathcal{A}) = \{1, 3\}$ *and* $BPO(\mathcal{A}) = \{1, 2\}$.

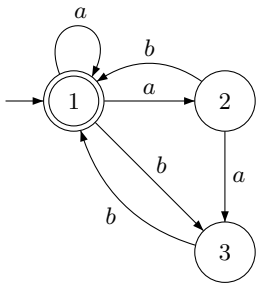

Figure 2.8: $\mathcal{A}$ monoidal automaton

Let us consider now, for each $i \geq 2$, the bpo's of an automaton in which there are i different edges going out from them.

Let $\mathcal{A} = (Q, I, F, \mathcal{F})$ be an automaton over A. For each state $x \in Q$, let m_x be the number of edges going out from x. It is easy to see that $|\mathcal{F}| - |Q| = \sum_{x \in Q}(m_x - 1)$. For each $i \geq 2$, let us consider the set

$$BPO_i(\mathcal{A}) = \{x \in Q \mid m_x = i\}$$

For simplicity of notation, we will moreover consider the sets

$$BPO_0(\mathcal{A}) = \{x \in Q \mid m_x = 0\}, \quad BPO_1(\mathcal{A}) = \{x \in Q \mid m_x = 1\}$$

When no confusion arises we will write BPO_i in place of $BPO_i(\mathcal{A})$.

Example 2.2.2 *In figure 2.9 we have* $\mathcal{A}$*, a monoidal automaton on* $A = \{a, b, c\}$ *with* $BPI(\mathcal{A}) = \{1\}$ *and* $BPO(\mathcal{A}) = \{1, 2\}$*. One has that* $BPO_1(\mathcal{A}) = \{3\}$*,* $BPO_2(\mathcal{A}) = \{1\}$ *and* $BPO_3(\mathcal{A}) = \{2\}$.

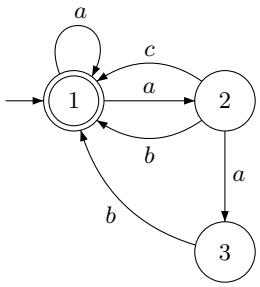

Figure 2.9: $\mathcal{A}$ monoidal automaton on $A = \{a, b, c\}$.

Let now

$$m_{\mathcal{A}} = max_{x \in Q}\{m_x\}$$

be the maximum number of edges going out from states of $\mathcal{A}$.

Since $Q = \cup_{i \geq 0} BPO_i$ and, for each $i \neq j$, $BPO_i \cap BPO_j = \emptyset$, then

$$|\mathcal{F}| - |Q| = \sum_{i=0,\ldots,m_{\mathcal{A}}} (\sum_{x \in BPO_i} (m_x - 1))$$

For each $i = 0, \ldots m_{\mathcal{A}}$, if $x \in BPO_i$ then $m_x - 1 = i - 1$ and we get

$$|\mathcal{F}| - |Q| = \sum_{i=0,\ldots,m_{\mathcal{A}}} |BPO_i|(i-1)$$

If $\mathcal{A}$ is a monoidal automaton then $BPO_0 = \emptyset$ as the following proposition implies:

Proposition 2.2.3 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton. One has $|\mathcal{F}| - |Q| = \sum_{i=2,\ldots,m_{\mathcal{A}}} |BPO_i|(i-1)$.*

Proof. Since, by hypothesis, $L(\mathcal{A}) \neq \{\varepsilon\}$ and $\mathcal{A}$ is trim with a unique final state equal to the initial one, then $1 \leq m_x \leq m_{\mathcal{A}}$. □

If, moreover, $\mathcal{A}$ is a deterministic monoidal automaton on the alphabet A of cardinality n it trivially follows that:

Corollary 2.2.4 *Let A be an alphabet of cardinality n. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a deterministic monoidal automaton. One has*

$$|\mathcal{F}| - |Q| = \sum_{i=2,\ldots,n} |BPO_i|(i-1)$$

In the example in figure 2.8 we have that $|\mathcal{F}| - |Q| = 6 - 3 = 3$ and

$$\sum_{i=2,\ldots,n} |BPO_i|(i-1) = |BPO_2| + 2|BPO_3| = 3$$

so the equality of corollary 2.2.4 holds.

2.2.2 Study of semiflower automata with at most one bpi

In this subsection we study semiflower automata with at most one bpi. In particular we characterize the submonoids recognized by unambiguous monoidal automata with no bpi and we find a general formula for the rank of submonoids recognized by semi-flower automata with a unique bpi depending on the characteristics of such automata.

Let us begin by giving a proposition that links, in a monoidal automaton, the existence of bpi's with the existence of bpo's.

Proposition 2.2.5 *Let $\mathcal{A}$ be a monoidal automaton. $BPI(\mathcal{A}) = \emptyset$ if and only if $BPO(\mathcal{A}) = \emptyset$.*

Proof. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton with $BPO(\mathcal{A}) \neq \emptyset$. Let so $x \in Q$ be a bpo. Let e_1 and e_2 be two edges starting at x and let us consider p a simple path from $f(e_1)$ to 1 and q a simple path from $f(e_2)$ to 1. Let us consider r the longest suffix path in common between e_1p and e_2q. let us distinguish two cases: when $x = 1$ and when $x \neq 1$.

Let $x = 1$. If p, q are null paths then 1 is a bpi. If p is not null and q is the null path then e_2 is not a suffix of e_1p since p is simple and so $i(r)$ is a bpi. If p is null and q is not null we find analogously a bpi. Finally if p and q are not null paths then e_1p and e_2q are not suffixes each other since p and q are simple paths ending at 1. As before $i(r)$ is a bpi and so in all cases it results $BPI(\mathcal{A}) \neq \emptyset$.

Let $x \neq 1$. Let us prove that e_1p and e_2q are not suffixes each other. If, by contradiction, e_1p is a proper suffix of e_2q then $e_2q = q'e_1p$, with q' not null and $f(q') = x$. Let s be the simple path from 1 to x. The path s is not a suffix path of q' since q is simple and $x \neq 1$. Analogously q' is not a suffix path of s since s is simple. So if r is the longest suffix path in common between s and q' then $i(r)$ is a bpi and so $BPI(\mathcal{A}) \neq \emptyset$.
The converse is proved in an analogue way. □

Let us begin now to study the semi-flower automata with at most one bpi.

In the following theorem we have a characterization of unambiguous monoidal automata with no bpi. In particular, such automata recognize cyclic submonoids.

Theorem 2.2.6 *Let $\mathcal{A}$ be a monoidal automaton. If $BPI(\mathcal{A}) = \emptyset$ then $L(\mathcal{A})$ is cyclic. Moreover, if $\mathcal{A}$ is unambiguous then, if $L(\mathcal{A})$ is cyclic then $BPI(\mathcal{A}) = \emptyset$.*

Proof. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton such that $BPI(\mathcal{A}) = \emptyset$ then, by prop. 2.2.5, $BPO(\mathcal{A}) = \emptyset$. If $|Q| = 1$ then trivially $L(\mathcal{A})$ is cyclic. Let $|Q| > 1$ and let $x \in Q$, $x \neq 1$. Let p be a simple path from 1 to x and q a simple path from x to 1 then the cycle pq is, in particular, simple in 1. It is the unique cycle that is simple in 1. In fact if there is another cycle that is simple in 1, let it be c, then considering r the

longest suffix path in common between pq and c we get that the initial state of r is a bpi, that is a contradiction. So if u is the label of pq then $L(\mathcal{A}) = \{u\}^*$.

Conversely let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be an unambiguous monoidal automaton such that $L(\mathcal{A})$ is cyclic. Then $\mathcal{A}$ is semi-flower. If $BPI(\mathcal{A}) \neq \emptyset$ then there exists $x \in BPO(\mathcal{A})$. It follows that there exist two different cycles that are simple in 1 and so, by the unambiguity of $\mathcal{A}$, two free generators for $L(\mathcal{A})$, that is a contradiction. □

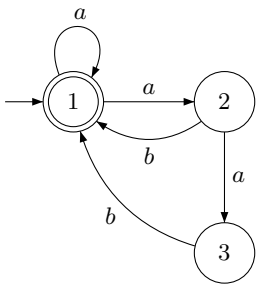

Figure 2.10: $\mathcal{A}$ monoidal automaton with $BPI(\mathcal{A}) = \{1\}$ that is a semi-flower automaton

It is interesting to note that if $\mathcal{A}$ is a monoidal automaton with initial-final state 1 such that $BPI(\mathcal{A}) = \{1\}$, then $\mathcal{A}$ is a semi-flower automaton, as explained in the following proposition. See, moreover, an example in figure 2.10

Proposition 2.2.7 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton such that $BPI(\mathcal{A}) = \{1\}$. It follows that $\mathcal{A}$ is a semi-flower automaton.*

Proof. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a monoidal automaton having $BPI(\mathcal{A}) = \{1\}$. Let c be a cycle in $\mathcal{A}$. If c is a cycle in 1 then, obviously, c visits 1. So let c be a cycle in $x \neq 1$. Let p be a simple path from 1 to x and let r be the longest suffix path in common between p and c. If p and c are not suffix each other then $i(r)$ is a bpi different from 1, that is a contradiction. Since p is a simple path then c cannot be a proper suffix of p, so p must be a proper suffix of c and c visits 1. □

In general, if $\mathcal{A}$ is a monoidal automaton such that $|BPI(\mathcal{A})| = 1$ then $\mathcal{A}$ is not necessarily a semi-flower automaton, as we can see in the example in figure 2.11.

An important class of semi-flower automata with a unique bpi is the class of literal automata, already defined in section 2.1.1. We have proved, in section 2.1.2, that such automata are deterministic semi-flower automata. We prove now that they have as unique bpi the initial-final state.

Proposition 2.2.8 *Let X be a finite prefix set and let $\mathcal{A}_X$ be the literal automaton of X^*, $\mathcal{A}_X = (Q, \varepsilon, \varepsilon, \delta)$. Then $\mathcal{A}_X$ is a deterministic semi-flower automaton with at most the state ε as bpi.*

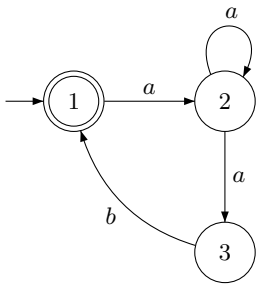

Figure 2.11: $\mathcal{A}$ monoidal automaton with a unique bpi that is not a semi-flower automaton

Proof. Let $\mathcal{A}_X = (Q, \varepsilon, \varepsilon, \delta)$. By prop. 2.1.12, $\mathcal{A}_X$ is a deterministic semi-flower automaton. We have so to prove that it has at most the state ε as bpi.
If it has no bpi's then it is done. Let us suppose that there exists at least one bpi. If there exists $u \in Q$, bpi for $\mathcal{A}_X$ such that $u \neq \varepsilon$, then in $\mathcal{A}_X$ there exist two different edges ending at u,

$$e_1 : v_1 \xrightarrow{a} u \quad and \quad e_2 : v_2 \xrightarrow{b} u$$

Since $u \neq \varepsilon$ it is $u = v_1 a = v_2 b$. Since a, b are letters in A it follows that $a = b$ then $v_1 = v_2$ and so $e_1 = e_2$ that is a contradiction. □

Example 2.2.9 *In figure 2.12, $\mathcal{A}_X$ is the literal automaton associated to X^*, with $X = \{bb, aba, abb, aaa\}$. It is a deterministic semi-flower automaton with ε as unique bpi.*

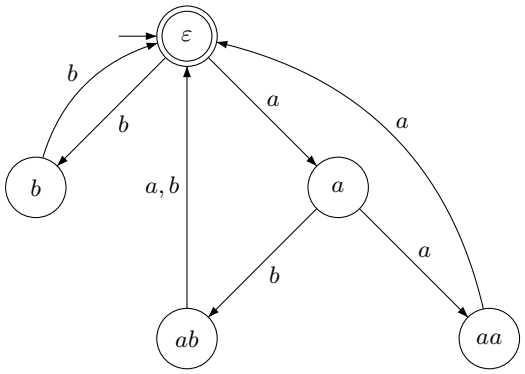

Figure 2.12: $\mathcal{A}_X$ literal automaton of X^*, where $X = \{bb, aba, abb, aaa\}$

Let us see now, given a semi-flower automaton with exactly one bpi, how to link the rank of the submonoid recognized by it with the characteristics of the automaton. The following theorem explains such a link:

Theorem 2.2.10 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a semi-flower automaton. If $|BPI(\mathcal{A})| = 1$ then $rk(L(\mathcal{A})) \leq |\mathcal{F}| - |Q| + 1$.*

Proof. Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a semi-flower automaton with a unique bpi. Let $BPI(\mathcal{A}) = \{x\}$. By prop. 2.1.3 and prop. 2.1.8, a set of generators for $L(\mathcal{A})$ is $Y_{\mathcal{A}}$, the finite set of labels of the cycles that are simple in 1. And by remark 2.1.7 these are simple cycles. So, we have to count the number of simple cycles with basis 1. For this purpose, let us prove that there is a bijection between $C_{\mathcal{A}}$, the set of simple cycles in 1, and $\mathcal{F}(x)$, the set of edges ending at x. Then we will have $rk(L(\mathcal{A})) \leq |Y_{\mathcal{A}}| \leq |C_{\mathcal{A}}| = |\mathcal{F}(x)|$.

Before making explicit such a bijection let us prove the following three points:

1.*If $c \in C_{\mathcal{A}}$ then c visits x.*

If $x = 1$ it is done. Let so $x \neq 1$ and let q be a simple path in $\mathcal{A}$ from x to 1. If we consider r the longest suffix path in common between q and c, we get that, if c does not visit x then the state $i(r)$ is a bpi different from x (the proof is analogous to that one of prop. 2.2.7).

2.*There is a unique simple path in $\mathcal{A}$ from x to* 1

$$q : x \longrightarrow 1$$

If $x = 1$ then the unique simple path from 1 to 1 is the null path. Let so $x \neq 1$. If there exist q and q' simple paths from x to 1, $q' \neq q$, then q and q' cannot be suffix each other since they are simple paths with the same initial vertex. So if we consider t the longest suffix path in common between q and q' we have that $i(t)$ is a bpi different from x, which is a contradiction.

3.*For each edge e ending at x there exists a unique simple path*

$$p_e : 1 \longrightarrow i(e)$$

If $x = 1$ and e is a self-loop in 1 then p_e is the null path. So let either $x \neq 1$ or e not a self-loop in 1.

If, by contradiction, there exists another simple path q_e from 1 to $i(e)$ then, considering as before t the longest suffix path in common between p_e and q_e, we get that $i(t)$ is a bpi different by x since p_e and q_e are simple paths.

Let us define now a map from $C_{\mathcal{A}}$ to $\mathcal{F}(x)$. Let

$$\varphi : C_{\mathcal{A}} \longrightarrow \mathcal{F}(x)$$

be such that

$$for\ each\ c \in C_{\mathcal{A}},\ \ \varphi(c) = e_c$$

where $e_c \in \mathcal{F}$ is the unique edge in c ending at x.

-φ is well defined since c is a simple cycle so it visits x only once.

-φ is injective. It follows from points 2 and 3.

-φ is surjective. For each $e \in \mathcal{F}(x)$ one has that $p_e eq$ is a simple cycle such that $\varphi(p_e eq) = e$.

Since φ is a bijection then:

$$|C_{\mathcal{A}}| = |\mathcal{F}(x)|$$

Let us consider the breadth-first search of $\mathcal{A}$ in 1 (see section 1.4). The BFS procedure visits each state of $\mathcal{A}$ reachable from 1. So, since $\mathcal{A}$ is monoidal, the BFS procedure visits each state of $\mathcal{A}$. Such procedure produces a tree, called the breadth-first search tree, $T = (Q, \mathcal{F}(T))$. In the following we investigate the edges of $\mathcal{A}$ that are not in T.

Let us consider two cases: when the unique bpi x is equal to 1 and when $x \neq 1$.

(1)Let $x = 1$.
Let us prove that

$$\mathcal{F} \setminus \mathcal{F}(T) = \mathcal{F}(1)$$

-$\mathcal{F} \setminus \mathcal{F}(T) \subseteq \mathcal{F}(1)$.

Let $e \in \mathcal{F} \setminus \mathcal{F}(T)$. Then either $f(e) = 1$ or $f(e)$ is visited by the BFS procedure by another edge of $\mathcal{A}$ ending at $f(e)$, let us call it g. Since $g \neq e$ then $f(e)$ is a bpi and $e \in \mathcal{F}(1)$.

-$\mathcal{F}(1) \subseteq \mathcal{F} \setminus \mathcal{F}(T)$ since the BFS procedure is applied in 1.

So, we have that

$$|\mathcal{F} \setminus \mathcal{F}(T)| = |\mathcal{F}(1)|$$

(2) Let $x \neq 1$.

For the BFS procedure there is a unique edge in T ending at x, let us call it e_x.

We will show now that all edges of $\mathcal{A}$ missing in T are those ones ending at x plus e_1, the end edge of q, minus e_x.

$$\mathcal{F} \setminus \mathcal{F}(T) = (\mathcal{F}(x) \cup \{e_1\}) \setminus \{e_x\}$$

-$\{e_1\} \subseteq \mathcal{F} \setminus \mathcal{F}(T)$ since the BFS procedure is applied in 1.

-$(\mathcal{F}(x) \setminus \{e_x\}) \subseteq \mathcal{F} \setminus \mathcal{F}(T)$. Let $f \in \mathcal{F}(x)$, $f \neq e_x$. As observed before, e_x is the only edge in T arriving in x and so $f \notin \mathcal{F}(T)$.

- $\mathcal{F} \setminus \mathcal{F}(T) \subseteq (\mathcal{F}(x) \cup \{e_1\}) \setminus \{e_x\}$. In fact, if $e \in \mathcal{F} \setminus \mathcal{F}(T)$ then $f(e) = 1$ and so $e = e_1$, or $f(e)$ is visited by the BFS procedure by another edge in $\mathcal{A}$ ending at $f(e)$, let us call it g. Since $g \neq e$ then $f(e)$ is a bpi and $e \in \mathcal{F}(x)$. Since g is in T then $g = e_x$ and so $e \neq e_x$.

So we have that

$$|\mathcal{F} \setminus \mathcal{F}(T)| = |(\mathcal{F}(x) \cup \{e_1\}) \setminus \{e_x\}| = |\mathcal{F}(x)|$$

So either $x = 1$ or $x \neq 1$ we get

$$|\mathcal{F} \setminus \mathcal{F}(T)| = |\mathcal{F}(x)| = |C_{\mathcal{A}}| \tag{2.2.1}$$

and by the properties of trees we have

$$|\mathcal{F}(T)| = |Q(T)| - 1 = |Q| - 1 \tag{2.2.2}$$

By 2.2.1 and by 2.2.2 one has

$$|C_{\mathcal{A}}| = |\mathcal{F} \setminus \mathcal{F}(T)| = |\mathcal{F}| - |Q| + 1$$

and so, by prop. 2.1.8,

$$rk(L(\mathcal{A})) \leq |C_{\mathcal{A}}| = |\mathcal{F}| - |Q| + 1$$

that concludes the proof. □

Example 2.2.11 *In figure 2.13 we have $\mathcal{A}$, a semi-flower automaton with a unique bpi with $BPI(\mathcal{A}) = \{3\}$ and $L(\mathcal{A}) = \{aab, abb, bb\}^*$. One has*

$$rk(L(\mathcal{A})) = 3$$

and

$$|\mathcal{F}| - |Q| + 1 = 5 - 3 + 1$$

The edges underlined are those ones of $T = (Q, \mathcal{F}(T))$ the BFS tree in 1.

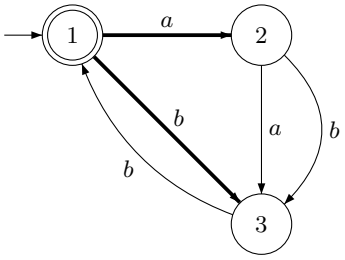

Figure 2.13: $\mathcal{A}$ semi-flower automaton with a unique bpi

If an automaton $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ has more than one bpi then it is not more true that $rk(L(\mathcal{A})) \leq |\mathcal{F}| - |Q| + 1$. Let us see an example:

Example 2.2.12 *In figure 2.14 $\mathcal{A}$ is a semi-flower automaton with more than one bpi,*

$$BPI(\mathcal{A}) = \{1, 2, 3\}$$

One has

$$L(\mathcal{A}) = \{aab, aaa, bab, baa, bb, ba\}^*$$

and

$$rk(L(\mathcal{A})) = 6 > |\mathcal{F}| - |Q| + 1 = 4$$

The edges underlined are those ones of $T = (Q, \mathcal{F}(T))$ a BFS tree in 1.

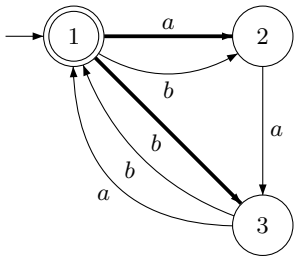

Figure 2.14: $\mathcal{A}$ semi-flower automaton with more than one bpi

Remark 2.2.13 *If an automaton $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ has a unique bpi then, taking a BFS tree T in 1, to each edge not contained in T one can associate a cycle that is simple in 1 and so a generator for $L(\mathcal{A})$. This is another possible proof of theorem 2.2.10.*

Using the notation of the proof of theorem 2.2.10 one has that, if $x = 1$ then $\mathcal{F} \setminus \mathcal{F}(T) = \mathcal{F}(1)$ and if $x \neq 1$ then $\mathcal{F} \setminus \mathcal{F}(T) = (\mathcal{F}(x) \cup \{e_1\}) \setminus \{e_x\}$. In particular, for each $e \in \mathcal{F} \setminus \mathcal{F}(T)$, p_e is the simple path in T from 1 to $i(e)$.
One can define the following map from the set of edges not contained in T to $C_{\mathcal{A}}$:

$$\psi : (\mathcal{F}(x) \cup \{e_1\}) \setminus \{e_x\} \longrightarrow C_{\mathcal{A}}$$

such that

$$\psi(e_1) = p_{e_x} e_x q$$

and for each $e \in \mathcal{F}(x) \setminus \{e_x\}$

$$\psi(e) = p_e e q$$

Such a map is well defined and injective since every cycle in $C_{\mathcal{A}}$ visits the bpi x just once and it is trivially surjective.

In the example in figure 2.11 one has:

$$\begin{cases} \psi(2 \xrightarrow{a} 3) = 1 \xrightarrow{a} 2 \xrightarrow{a} 3 \xrightarrow{b} 1 \\ \psi(2 \xrightarrow{b} 3) = 1 \xrightarrow{a} 2 \xrightarrow{b} 3 \xrightarrow{b} 1 \\ \psi(1 \xrightarrow{b} 3) = 1 \xrightarrow{b} 3 \xrightarrow{b} 1 \end{cases}$$

Using the same arguments, if $\mathcal{A}$ is a semi-flower automaton with more than one bpi to each edge not contained in T one can associate more cycles since

-there can be different paths from a bpi to 1

-there can be different simple paths from 1 *to an edge ending in a bpi*

For example in figure 2.14 the paths $1 \xrightarrow{a} 2 \xrightarrow{a} 3 \xrightarrow{a} 1$ *and* $1 \xrightarrow{a} 2 \xrightarrow{a} 3 \xrightarrow{b} 1$ *correspond to the edge* $2 \xrightarrow{a} 3$.

It follows from the proof of theorem 2.2.10 that in the unambiguous case the following holds:

Proposition 2.2.14 *If $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ is an unambiguous semi-flower automaton such that $|BPI(\mathcal{A})| = 1$ then $rk(L(\mathcal{A})) = |\mathcal{F}| - |Q| + 1$.*

We will see that a similar result holds for free groups.

Let us give now a theorem that links the rank of the submonoid generated by a given semi-flower automaton with a unique bpi with the cardinality of the sets of BPO_i, introduced in the subsection 2.2.1. It is a consequence of theorem 2.2.10 and of proposition 2.2.3.

Theorem 2.2.15 *Let A be an alphabet of cardinality n. Let $\mathcal{A}$ be a semi-flower automaton on A with a unique bpi then*

$$rk(L(\mathcal{A})) \leq \sum_{i=2,\ldots,m_{\mathcal{A}}} |BPO_i|(i-1) + 1$$

Moreover, if $\mathcal{A}$ is unambiguous then it follows the equality and, if $\mathcal{A}$ is deterministic then

$$rk(L(\mathcal{A})) = \sum_{i=2,\ldots,n} |BPO_i|(i-1) + 1$$

2.3 The intersection of two submonoids: the prefix case

The study of the intersection of two submonoids of finite rank is not trivial at all. In fact, by a result of Latteux and Leguy ([17]), every language is regular if and only if the submonoid generated by it is obtained as omomorphic image of the intersection of two finitely generated monoids:

Theorem 2.3.1 *Let A be an alphabet and R a language of A^*. Then R is a regular language if and only if there exist two finite languages F_1, F_2 and a morphism g such that $R^* = g({F_1}^* \cap {F_2}^*)$.*

In this section we investigate the intersection of two submonoids of A^* generated by finite prefix sets by studying the product of two deterministic semi-flower automata recognizing the two given submonoids.

In general, given H and K finitely generated submonoids we say that H and K satisfy the *Hanna Neumann property* if they satisfy the following inequality:

$$\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$$

So we pose the problem of finding H and K satisfying the Hanna Neumann property. In the subsection 2.3.2 we prove that if H and K are submonoids generated by finite prefix sets such that their product automaton has at most one bpi then H and K satisfy the Hanna Neumann property. While in subsection 2.3.3 we furnish a class of examples of H and K with finite intersection's rank not satisfying this property.

2.3.1 Intersection of two submonoids: definitions and properties of the product automaton

In this subsection we recall the definition and some properties of the product automaton. We moreover give a proposition that links the bpo's in the product automaton of two deterministic automata with the bpo's of the starting automata. For the definitions and the properties of the product automaton we recall [12].

Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two automata

$$\mathcal{A}_1 = (Q_1, I_1, F_1, \mathcal{F}_1), \quad \mathcal{A}_2 = (Q_2, I_2, F_2, \mathcal{F}_2)$$

The product automaton is defined as

$$\mathcal{A}_1 \times \mathcal{A}_2 = (Q_1 \times Q_2, I_1 \times I_2, F_1 \times F_2, \mathcal{F})$$

with

$$\mathcal{F} = \{((p_1, p_2), a, (q_1, q_2)),\ a \in A \mid (p_1, a, q_1) \in \mathcal{F}_1 \text{ and } (p_2, a, q_2) \in \mathcal{F}_2\}$$

Let us consider in $\mathcal{A}_1 \times \mathcal{A}_2$ only the accessible and coaccessible states. One has that the language recognized by $\mathcal{A}_1 \times \mathcal{A}_2$ is the intersection of the languages recognized by $\mathcal{A}_1$ and $\mathcal{A}_2$, that is

$$L(\mathcal{A}_1 \times \mathcal{A}_2) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$$

In particular, the product of two automata recognizing submonoids recognizes the intersection of such submonoids.

Some properties of two given automata, $\mathcal{A}_1$ and $\mathcal{A}_2$, are maintained in the product $\mathcal{A}_1 \times \mathcal{A}_2$ as shown in the following proposition:

Proposition 2.3.2 *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two automata.*

1. *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are monoidal automata then $\mathcal{A}_1 \times \mathcal{A}_2$ is a monoidal automaton.*
2. *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are unambiguous automata then $\mathcal{A}_1 \times \mathcal{A}_2$ is an unambiguous automaton.*
3. *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are deterministic automata then $\mathcal{A}_1 \times \mathcal{A}_2$ is a deterministic automaton.*

On the other hand the product of two semi-flower automata is not necessarily a semi-flower automaton as it is shown in the following example. This is in agreement with the fact that the intersection of two finitely generated submonoids is not necessarily finitely generated.

Example 2.3.3 *In figure 2.15 there are the literal automata associated to $H = \{aab, aba\}^*$ and $K = \{a, baaba\}^*$, let us denote them by $\mathcal{A}_H$ and $\mathcal{A}_K$. These are, in particular, deterministic semi-flower automata. In figure 2.16 there is their product $\mathcal{A}_H \times \mathcal{A}_K$ that is not a semi-flower automaton. As we can note $H \cap K = \{a(abaaba)^*baaba\}^*$ is not finitely generated.*

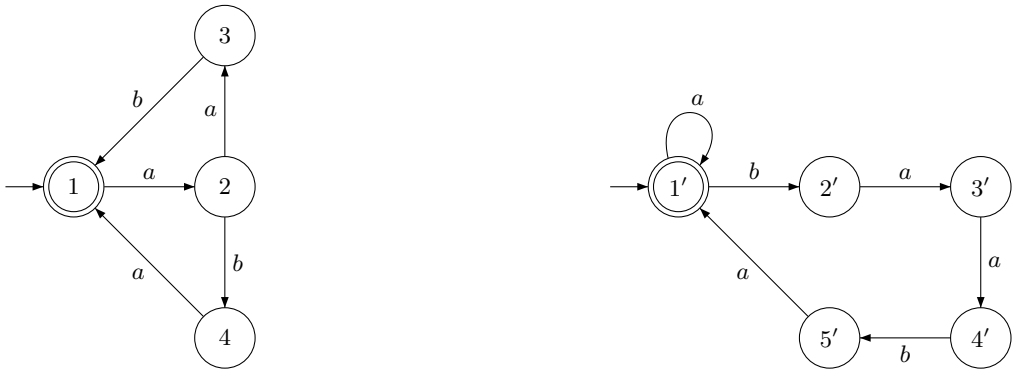

Figure 2.15: $\mathcal{A}_H$ and $\mathcal{A}_K$ literal automata associated to $H = \{aab, aba\}^*$ and $K = \{a, baaba\}^*$.

Moreover, if the product of two semi-flower automata with a unique bpi is a semi-flower automaton, then it is not necessarily a semi-flower automaton with a unique bpi (see the following example).

Example 2.3.4 *In figure 2.17 there are $\mathcal{A}_H$ and $\mathcal{A}_K$ the literal automata associated to $H = \{b, aa, ab\}^*$ and $K = \{a, bb, baa, bab\}^*$ and their product automaton. One has that $\mathcal{A}_H$ and $\mathcal{A}_K$ are deterministic semi-flower automata with a unique bpi while $\mathcal{A}_H \times \mathcal{A}_K$ is a deterministic semi-flower automaton with more than one bpi.*

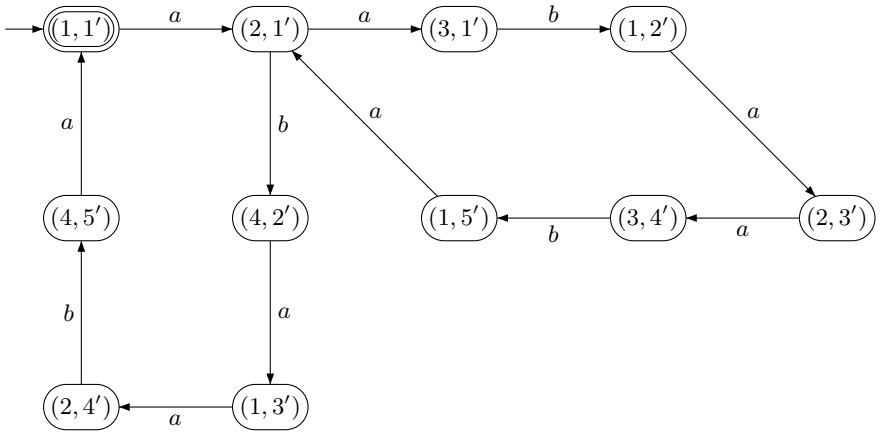

Figure 2.16: $\mathcal{A}_H \times \mathcal{A}_K$ not semi-flower automata with $\mathcal{A}_H$ and $\mathcal{A}_K$ literal automata associated to $H = \{aab, aba\}^*$ and $K = \{a, baaba\}^*$.

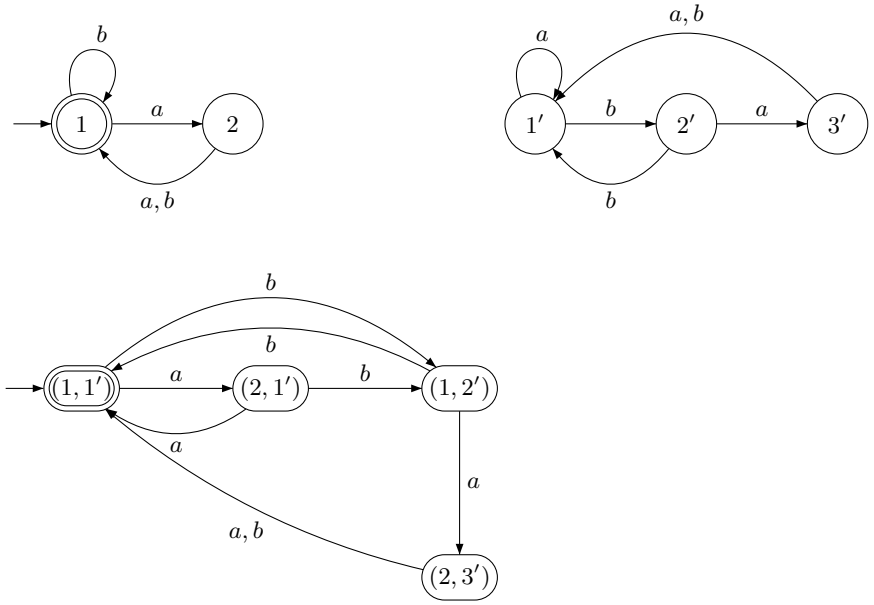

Figure 2.17: $\mathcal{A}_H$, $\mathcal{A}_K$, literal automata associated to $H = \{b, aa, ab\}^*$ and $K = \{a, bb, baa, bab\}^*$, and $\mathcal{A}_H \times \mathcal{A}_K$.

Let now $\mathcal{A}_1$ and $\mathcal{A}_2$ be two deterministic automata on A

$$\mathcal{A}_1 = (Q_1, x_1, F_1, \delta_1),\ \mathcal{A}_2 = (Q_2, y_1, F_2, \delta_2)$$

The product automaton is defined as

$$\mathcal{A}_1 \times \mathcal{A}_2 = (Q_1 \times Q_2, (x_1, y_1), F_1 \times F_2, \delta)$$

where for each $(x, y) \in Q_1 \times Q_2$, for each $a \in A$

$$\delta((x, y), a) = (\delta_1(x, a), \delta_2(y, a))$$

Remark 2.3.5 *From now on, let us consider an alphabet A of cardinality n.*

Let us give now a lemma that links the bpo's in the product of two deterministic automata on A with the bpo's of the respective automata.

Lemma 2.3.6 *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two deterministic automata on A. One has*

$$BPO_t(\mathcal{A}_1 \times \mathcal{A}_2) \subseteq \cup_{t \leq r,s \leq n}(BPO_r(\mathcal{A}_1) \times BPO_s(\mathcal{A}_2))$$

Proof. Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two deterministic automata on $A = \{a_1, \ldots, a_n\}$, with $\mathcal{A}_1 = (Q_1, x_1, F_1, \delta_1)$ and $\mathcal{A}_2 = (Q_2, y_1, F_2, \delta_2)$.

If $(x, y) \in BPO_t(\mathcal{A}_1 \times \mathcal{A}_2)$ then there exist t different edges going out from (x, y). So, for each $i = 1, \ldots, t$, there exists $(x_i, y_i) \in Q_1 \times Q_2$, $a_i \in A$ such that

$$\delta((x, y), a_i) = (x_i, y_i)$$

And, since $\mathcal{A}_1 \times \mathcal{A}_2$ is deterministic, for $i \neq j$ $a_i \neq a_j$.
By definition, for each $i = 1, \ldots, t$

$$\delta((x, y), a_i) = (\delta(x, a_i), \delta(y, a_i)) = (x_i, y_i)$$

Then for each $i = 1, \ldots, t$,

$$\delta(x, a_i) = x_i, \quad \delta(y, a_i) = y_i$$

Since $a_i \neq a_j$, for each i, j with $i \neq j$, then there are at least t different edges going out from x and so $x \in BPO_r(\mathcal{A}_1)$ with $r \geq t$. Analogously $y \in BPO_s(\mathcal{A}_2)$ with $s \geq t$ and so the thesis follows. □

Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be deterministic automata on A. Let $c_i = |BPO_i(\mathcal{A}_1)|$ and $d_i = |BPO_i(\mathcal{A}_2)|$, for each $i = 1, \ldots, n$. In terms of cardinality of the sets of bpo's the lemma 2.3.6 becomes:

Corollary 2.3.7 $|BPO_t(\mathcal{A}_1 \times \mathcal{A}_2)| \leq \sum_{t \leq r,s \leq n} c_r d_s$.

Example 2.3.8 *In the example in figure 2.17 one has that*

$$BPO_2(\mathcal{A}_1 \times \mathcal{A}_2) = \{(1,1'),(2,1'),(1,2'),(2,3')\}$$

$$BPO_2(\mathcal{A}_1) = \{1,2\}, \quad BPO_2(\mathcal{A}_2) = \{1',2',3'\}$$

and

$$BPO_2(\mathcal{A}_1) \times BPO_2(\mathcal{A}_2) = \{(1,1'),(1,2'),(1,3'),(2,1'),(2,2'),(2,1')\}$$

So it results

$$BPO_2(\mathcal{A}_1 \times \mathcal{A}_2) \subseteq BPO_2(\mathcal{A}_1) \times BPO_2(\mathcal{A}_2)$$

and

$$|BPO_2(\mathcal{A}_1 \times \mathcal{A}_2)| = 4 \leq |BPO_2(\mathcal{A}_1)||BPO_2(\mathcal{A}_1)| = 6$$

Let us give now a lemma that will allow us to prove the inequality of Hanna Neumann for submonoids recognized by deterministic semi-flower automata with a unique bpi such that their product automaton has at most one bpi. Let $C = \{c_1, \ldots, c_n\}$ and $D = \{d_1, \ldots, d_n\}$ be two sets of natural numbers $\mathbb{N}$.

Let

$$P_{CD} = \sum_{t=2,\ldots,n} (t-1)(\sum_{t \leq r \leq n} c_r \sum_{t \leq s \leq n} d_s)$$

and

$$Q_{CD} = (\sum_{i=2,\ldots,n} (i-1)c_i)(\sum_{j=2,\ldots,n} (j-1)d_j)$$

Lemma 2.3.9 *Let $C = \{c_1, \ldots, c_n\}$ and $D = \{d_1, \ldots, d_n\}$ be subsets of $\mathbb{N}$. One has $P_{CD} \leq Q_{CD}$. Moreover if there exist $k > 2$ and $l \geq 2$ such that either $c_k \neq 0$, $d_l \neq 0$ or $c_l \neq 0$, $d_k \neq 0$ then $P_{CD} < Q_{CD}$.*

Proof. Let us write P_{CD} and Q_{CD} as sum of elements $c_k d_l$ and let α_{kl} (resp. β_{kl}) be the natural number correspondent to the number of times $c_k d_l$ appears in P_{CD} (resp. Q_{CD}). Then we have:

$$P_{CD} = \sum_{2 \leq k,l \leq n} \alpha_{kl} c_k d_l$$

and

$$Q_{CD} = \sum_{2 \leq k,l \leq n} \beta_{kl} c_k d_l$$

In the following we calculate, for each $k, l \geq 2$, α_{kl} and β_{kl} and, in particular, we prove that $\alpha_{kl} \leq \beta_{kl}$.

Let us write $P_{CD} = P_2 + 2P_3 + \ldots + (n-1)P_n$ where $P_t = P_t{}' P_t{}''$ with $P_t{}' = \sum_{t \leq r \leq n} c_r$ and $P_t{}'' = \sum_{t \leq s \leq n} d_s$.

Let $k, l \in \{2, \ldots, n\}$. Then c_k appears in P_i' if and only if $i \leq k$ and d_l appears in P_i'' if and only if $i \leq l$. Then $c_k d_l$ appears in $P_i' P_i''$ if and only if $i = 1, \ldots, \overline{k}$ where $\overline{k} = min(k, l)$. So $c_k d_l$ appears in P_{CD} with coefficient

$$\alpha_{kl} = 1 + 2 + \ldots + (\overline{k} - 1)$$

Let $Q = Q'Q''$ with $Q' = \sum_{i=2,\ldots,n}(i-1)c_i$ and $Q'' = \sum_{i=2,\ldots,n}(i-1)d_i$. Let $k, l \in \{2, \ldots, n\}$. Then c_k appears in Q' with coefficient $k-1$ and d_l appears in Q'' with coefficient $l-1$ so

$$\beta_{kl} = (k-1)(l-1)$$

It is $1+2+\ldots+(\overline{k}-1) < (\overline{k}-1)(\overline{k}-1)$ for $\overline{k} > 2$ and $1+2+\ldots+(\overline{k}-1) = (\overline{k}-1)(\overline{k}-1)$ for $\overline{k} = 2$.

So $\alpha_{kl} \leq \beta_{kl}$, for each $k, l \geq 2$. In particular $\alpha_{kl} < \beta_{kl}$ for $\overline{k} > 2$. So we get that $P_{CD} \leq Q_{CD}$.

Let us suppose that there exist $k > 2$ and $l \geq 2$ such that $c_k \neq 0$ and $d_l \neq 0$. If $\overline{k} = 2$ then $\alpha_{kl} = 1 + 2 + \ldots + (\overline{k} - 1) = (\overline{k} - 1)(\overline{k} - 1) < (k-1)(l-1) = \beta_{kl}$. So $\alpha_{kl} < \beta_{kl}$, for $\overline{k} \geq 2$ and since $c_k d_l \neq 0$ then $P_{CD} < Q_{CD}$.
It is proved analogously if there exist $k > 2$ and $l \geq 2$ such that $c_l \neq 0$ and $d_k \neq 0$. □

Example 2.3.10 *Let $C = \{c_1, c_2, c_3, c_4\}$ and $D = \{d_1, d_2, d_3, d_4\}$. One has:*

$P_{CD} = (c_2 + c_3 + c_4)(d_2 + d_3 + d_4) + 2(c_3 + c_4)(d_3 + d_4) + 3c_4 d_4 = c_2 d_2 + c_2 d_3 + c_2 d_4 + c_3 d_2 + c_3 d_3 + c_3 d_4 + c_4 d_2 + c_4 d_3 + c_4 d_4 + 2c_3 d_3 + 2c_3 d_4 + 2c_4 d_3 + 2c_4 d_4 + 3c_4 d_4 = c_2 d_2 + c_2 d_3 + c_2 d_4 + c_3 d_2 + 3c_3 d_3 + 3c_3 d_4 + c_4 d_2 + 3c_4 d_3 + 6c_4 d_4$

$Q_{CD} = (c_2 + 2c_3 + 3c_4)(d_2 + 2d_3 + 3d_4) = c_2 d_2 + 2c_2 d_3 + 3c_2 d_4 + 2c_3 d_2 + 4c_3 d_3 + 6c_3 d_4 + 3c_4 d_2 + 6c_4 d_3 + 9c_4 d_4$

2.3.2 Intersection of two submonoids finitely generated by prefix codes: product automaton with a unique bpi

In this subsection we consider the case of the intersection of two submonoids finitely generated by prefix sets such that there exist two deterministic semi-flower automata with a unique bpi whose product has at most one bpi. In this case we prove that the Hanna Neumann property holds.

Let now H and K be submonoids of A^* finitely generated by prefix sets of cardinality at least two. Let $\mathcal{A}_H$ and $\mathcal{A}_K$ be the literal automata associated to H and K respectively. As we have seen such automata are deterministic semi-flower with a unique bpi.

Since $\mathcal{A}_H$ and $\mathcal{A}_K$ are deterministic monoidal automata then $\mathcal{A}_H \times \mathcal{A}_K$ is still a deterministic monoidal automaton.

Remark 2.3.11 *Let $\mathcal{A}_H \times \mathcal{A}_K$ be a semi-flower automaton. If it has not bpi then $\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$. In fact, if $H \cap K = \{\varepsilon\}$ then $\widetilde{rk}(H \cap K) = 0$ otherwise by prop.2.2.6, $H \cap K$ is cyclic and so $\widetilde{rk}(H \cap K) = 0$.*

If $\mathcal{A}_H \times \mathcal{A}_K$ is a semi-flower automaton with a unique bpi we get the Hanna Neumann inequality, as it is stated in the following theorem. Moreover, if there exist either bpo's with at least three edges going out from them in $\mathcal{A}_H$ and at least one bpo in $\mathcal{A}_K$ or bpo's with at least three edges going out from them in $\mathcal{A}_K$ and at least one bpo in $\mathcal{A}_H$ then $\widetilde{rk}(H \cap K) < \widetilde{rk}(H)\widetilde{rk}(K)$.

Theorem 2.3.12 *Let H and K be submonoids generated by finite prefix sets. If there exist $\mathcal{A}_H$ and $\mathcal{A}_K$ deterministic semi-flower automata with a unique bpi such that $\mathcal{A}_H \times \mathcal{A}_K$ is a semi-flower automaton with a unique bpi then*

$$\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$$

Moreover, letting $c_i = |BPO_i(\mathcal{A}_H)|$ and $d_i = |BPO_i(\mathcal{A}_K)|$, for each $i = 1, \ldots, n$, the strict inequality holds if there exist $i > 2$, $j \geq 2$ such that either $c_i \neq 0$, $d_j \neq 0$ or $c_j \neq 0$, $d_i \neq 0$.

Proof. Let H and K be submonoids generated by finite prefix sets and let $\mathcal{A}_H$ and $\mathcal{A}_K$ be deterministic semi-flower automata with a unique bpi such that their product $\mathcal{A}_H \times \mathcal{A}_K$ is deterministic semi-flower automaton with a unique bpi. Applying cor.2.3.7 and theorem 2.2.15 we get:

$$\widetilde{rk}(H \cap K) \leq \sum_{t=2\ldots n} (t-1)\Big(\sum_{t \leq r \leq n} c_r \sum_{t \leq s \leq n} d_s\Big)$$

On the other hand by theorem 2.2.15 it is

$$\widetilde{rk}(H)\widetilde{rk}(K) = \Big(\sum_{i=2,\ldots n} (i-1)c_i\Big)\Big(\sum_{j=2,\ldots n} (j-1)d_j\Big)$$

Applying lemma 2.3.9 to the sets $C = \{c_1, \ldots, c_n\}$ and $D = \{d_1, \ldots, d_n\}$ we get that

$$\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$$

Let us suppose that there exist $i > 2$, $j \geq 2$ such that either $c_i \neq 0$, $d_j \neq 0$ or $c_j \neq 0$, $d_i \neq 0$. If $c_i \neq 0$ and $d_j \neq 0$ then $BPO_i(\mathcal{A}_H) \neq \emptyset$ and $BPO_j(\mathcal{A}_K) \neq \emptyset$. Then, by lemma 2.3.9, we get that

$$\widetilde{rk}(H \cap K) < \widetilde{rk}(H)\widetilde{rk}(K)$$

It is proved analogously if there exist $i > 2$, $j \geq 2$ such that $c_j \neq 0$, $d_i \neq 0$. □

Example 2.3.13 *Let $H = \{ba, bb, aa\}^*$ and $K = \{baa, bbb\}^*$. The submonoids H and K are generated by prefix sets. Let $\mathcal{A}_H$ and $\mathcal{A}_K$ be the literal automata associated to them (see figure 2.18). One has that $\mathcal{A}_H \times \mathcal{A}_K$ is an automaton with a unique bpi (see figure 2.19). A minimal set of generators of $H \cap K$ is the set of labels of the cycles that are simple in $(1, 1')$, that is $Y_{\mathcal{A}} = \{b^6, b^4a^2\}$. The reduced rank of $H \cap K$ is $\widetilde{rk}(H \cap K) = 1$. On the other hand $\widetilde{rk}(H)\widetilde{rk}(K) = 2$ so it results $\widetilde{rk}(H \cap K) < \widetilde{rk}(H)\widetilde{rk}(K)$.*

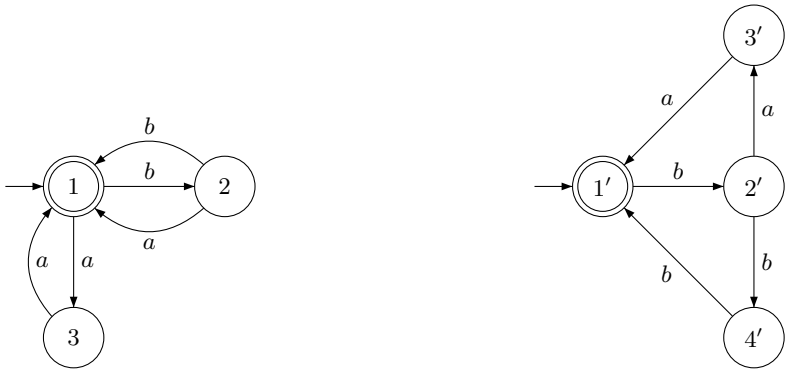

Figure 2.18: $\mathcal{A}_H$, $\mathcal{A}_K$ literal automata of $H = \{ba, bb, aa\}^*$ and $K = \{baa, bbb\}^*$ respectively.

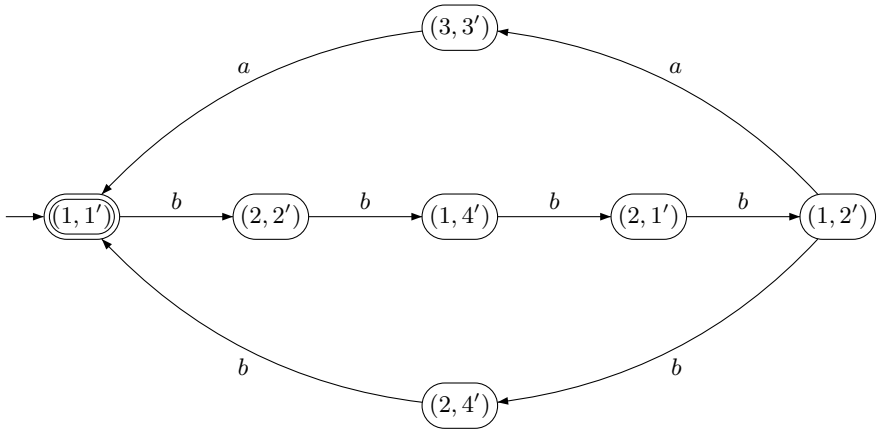

Figure 2.19: $\mathcal{A}_H \times \mathcal{A}_K$, where $\mathcal{A}_H$ and $\mathcal{A}_K$ are the literal automata of $H = \{ba, bb, aa\}^*$ and $K = \{baa, bbb\}^*$

The previous example shows that there exist H and K with $\mathcal{A}_H$, $\mathcal{A}_K$ and $\mathcal{A}_H \times \mathcal{A}_K$ semi-flower with a unique bpi not having bpo's with more than two edges going out and, such that $\widetilde{rk}(H \cap K) < \widetilde{rk}(H)\widetilde{rk}(K)$. So the reverse of the second part of theorem 2.3.12 does not hold in general.

2.3.3 Intersection of two submonoids finitely generated by prefix codes: product automaton with more than one bpi

If H and K are submonoids finitely generated by prefix sets such that $\mathcal{A}_H \times \mathcal{A}_K$ is a deterministic semi-flower automaton with more than one bpi then it is not more true that $\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$.

There is a family of examples such that $\widetilde{rk}(H \cap K) > \widetilde{rk}(H)\widetilde{rk}(K)$ and, in particular, $rk(H \cap K) = 2^{log_2(rk(H))log_2(rk(K))}$:

Example 2.3.14 *Let p and q be two positive co-prime integers greater or equal to 2, $p, q \geq 2$. Let A be a binary alphabet and let $H = \{A^p\}^*$ and $K = \{A^q\}^*$, the set of words of length multiple of p and q respectively.*

One has $rk(H) = 2^p$ and so $p = log_2(rk(H))$. It is $H \cap K = \{A^{pq}\}^*$ and

$$rk(H \cap K) = 2^{pq} = 2^{log_2(rk(H))log_2(rk(K))}$$

In particular

$$\widetilde{rk}(H \cap K) = 2^{pq} - 1$$

and

$$\widetilde{rk}(H)\widetilde{rk}(K) = (2^p - 1)(2^q - 1) = 2^{p+q} - 2^p - 2^q + 1$$

Since $2^{pq} - 1 > 2^{p+q} - 2^p - 2^q + 1$ we get

$$\widetilde{rk}(H \cap K) > \widetilde{rk}(H)\widetilde{rk}(K)$$

See examples in figures 2.20 and 2.21. In figure 2.21, for simplicity of the picture, we have written two copies of the same vertex $(1, 1')$ but it must be considered the same vertex.

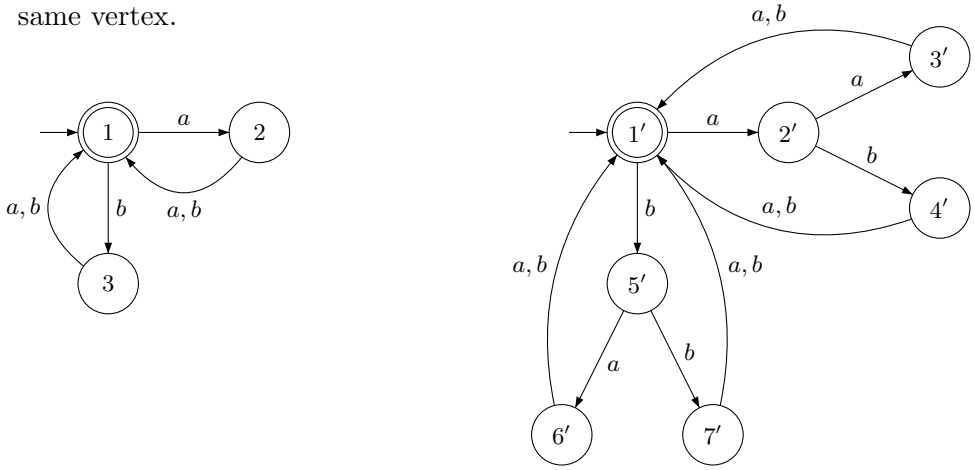

Figure 2.20: $\mathcal{A}_{A^{2*}}$, $\mathcal{A}_{A^{3*}}$, with A^2 the set of words in $A = \{a, b\}$ of length 2 and A^3 the set of words in $A = \{a, b\}$ of length 3

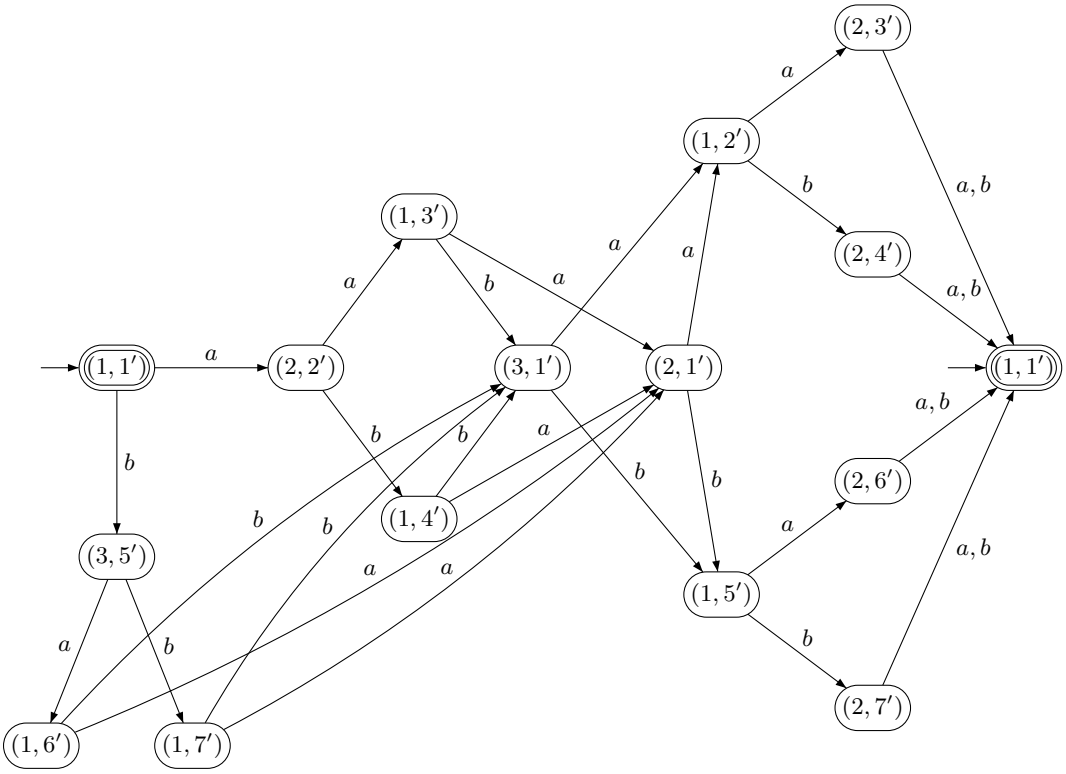

Figure 2.21: $\mathcal{A}_{A^2} \times \mathcal{A}_{A^3}$. This automaton has more than one bpi: $|BPI| = 5$. One has $\widetilde{rk}(H \cap K) = (2^6 - 1) = 63 > \widetilde{rk}(H)\widetilde{rk}(K) = (2^2 - 1)(2^3 - 1) = 21$

However, if $\mathcal{A}_H$ and $\mathcal{A}_K$ are semi-flower automata with a unique bpi such that their product has more than one bpi, then it is not necessarily $\widetilde{rk}(H \cap K) > \widetilde{rk}(H)\widetilde{rk}(K)$, as stated in the following example:

Example 2.3.15 *In figure 2.22 $\mathcal{A}_H$ and $\mathcal{A}_K$ are semi-flower automata with a unique bpi and $\mathcal{A}_H \times \mathcal{A}_K$ is a semi-flower automaton with more than one bpi.*

$$H = \{b, aa, ab\}^*$$

$$K = \{a, bb, baa, bab\}^*$$

and

$$H \cap K = \{aa, abaa, abab, abb, bb, baa, bab\}$$

Even if $\mathcal{A}_H \times \mathcal{A}_K$ has more than one bpi, the inequality of Hanna-Neumann holds:

$$\widetilde{rk}(H \cap K) = 6 = \widetilde{rk}(H)\widetilde{rk}(K) = 6$$

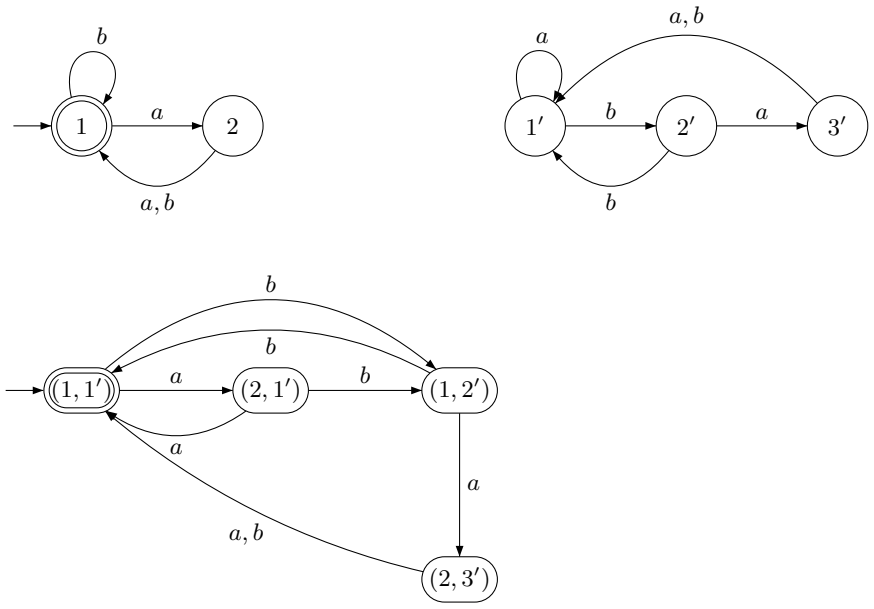

Figure 2.22: $\mathcal{A}_H$ and $\mathcal{A}_K$ semi-flower automata with a unique bpi and $\mathcal{A}_H \times \mathcal{A}_K$ semi-flower automata with more than one bpi.

2.4 The prefix case with two generators

If H and K are submonoids generated by prefix sets of two elements of A^* we get the result of Karhumäki (cf.[16]). In particular we prove that there exist two deterministic semi-flower automata recognizing H and K respectively and such that their product automaton has at most one bpi, so in the case of finite rank of the intersection we obtain that the rank of the intersection is less or equal to 2.

Let H be a submonoid of A^* generated by a finite prefix set of two elements X. Let $\mathcal{A}_X$ be the literal automaton of X^*, let us call it $\mathcal{A}_H$ for simplicity of notation. In the beginning we prove that $\mathcal{A}_H$ has a unique bpo with two edges starting at it and all the other vertices have just one edge starting at them.

Let us remember that if $\mathcal{A} = (Q, I, F, \mathcal{F})$ is an automaton over A then, for each state $x \in Q$, we denote by m_x be the number of edges going out from x.

Proposition 2.4.1 *Let H be a submonoid generated by a prefix set of two elements. Let $\mathcal{A}_H = (Q, 1, 1, \delta)$ be the literal automaton of H.*

For each $x \in Q$, it is $1 \leq m_x \leq 2$. Moreover there is a unique vertex $y \in Q$ such that $m_y = 2$.

Proof. Let H be a submonoid finitely generated by a prefix set of two elements $X = \{u, v\}$. Let $\mathcal{A}_H = (Q, 1, 1, \delta)$ be the literal automaton of H. Since $\mathcal{A}_H$ is monoidal then $1 \leq m_x$. Let $Q = X(A^+)^{-1} = \{u_1, \ldots, u_n\}$.

Let us observe that, for each $u_i \in Q$, there exists an edge starting at u_i with label a if and only if $u_i a$ is a prefix of u or a prefix of v.

Let $\overline{u}$ be the longest prefix in common between u and v, $u = \overline{u}u_1$ and $v = \overline{u}v_1$ with $u_1 \in aA^*$, $v_1 \in bA^*$, $a, b \in A$ and $a \neq b$. The vertex $\overline{u}$ is a bpo for $\mathcal{A}_H$ since $\overline{u}a$ is a prefix of u and $\overline{u}b$ is a prefix of v and $a \neq b$. So $m_{\overline{u}} = 2$.

Let now $w \in Q$, $w \neq \overline{u}$. If $|w| < |\overline{u}|$ then w is a proper prefix of $\overline{u}$ and so $\overline{u} = wu'$ with $u' \in cA^*$, $c \in A$. So $e : w \xrightarrow{c} wc$ (or $e : w \xrightarrow{c} 1$ if $wc = \varepsilon$) is the unique edge starting from w and we get $m_w = 1$. If $|w| > |\overline{u}|$ then either w is a prefix of u or a prefix of v, it cannot be a prefix of both since $\overline{u}$ is the longest prefix in common between u and v. So there is a unique edge starting from w and $m_w = 1$. □

For the product automaton of two literal automata, associated to submonoids generated by prefix sets of two elements, it holds the following:

Lemma 2.4.2 *Let H and K be submonoids generated by prefix sets of two elements such that $H \cap K \neq \{\varepsilon\}$. Let $\mathcal{A}_H \times \mathcal{A}_K = (Q, 1, 1, \delta)$.*

For each $w \in Q$, it is $1 \leq m_w \leq 2$. Moreover there is at most one vertex $z \in Q$ such that $m_z = 2$.

Proof. Since $\mathcal{A}_H \times \mathcal{A}_K$ is monoidal and $H \cap K \neq \{\varepsilon\}$ then, for each $x \in Q$, it is $m_x \geq 1$. Let $\mathcal{A} = \mathcal{A}_H \times \mathcal{A}_K$. Let $c_i = |BPO_i(\mathcal{A}_H)|$ and $d_i = |BPO_i(\mathcal{A}_K)|$ for each $i = 1, \ldots, n$. By cor. 2.3.7 we have that for each $t \geq 0$

$$|BPO_t(\mathcal{A})| \leq \sum_{t \leq r, s \leq n} c_r d_s$$

By prop. 2.4.1 we get $c_2 = 1$, $d_2 = 1$ and for each $s > 2$ $c_s = 0$ and $d_s = 0$.

For each $t > 2$ we obtain $|BPO_t(\mathcal{A})| \leq 0$ and so, for each $x \in Q$, $m_x \leq 2$. For $t = 2$ we get $|BPO_2(\mathcal{A})| \leq c_2 d_2 = 1$ and so there is at most one $x \in Q$ such that $m_x = 2$. □

We can now prove the result of Karhumäki (cf.[16]):

Theorem 2.4.3 *Let H and K be submonoids generated by prefix sets of two elements.*

1. *If $H \cap K$ is finitely generated then $rk(H \cap K)$ is at most two.*
2. *If $H \cap K$ is not finitely generated then there exist $u, v, w \in A^*$ such that $H \cap K = (u(v)^*w)^*$.*

Proof. Let H and K be submonoids generated by prefix sets of two elements and $\mathcal{A}_H$, $\mathcal{A}_K$ be the literal automata associated to H and K respectively. We denote $\mathcal{A} = \mathcal{A}_H \times \mathcal{A}_K$.

(1)*Let $rk(H \cap K) < \infty$*

Since $\mathcal{A}$ is a deterministic monoidal automaton recognizing $H \cap K$, that is finitely generated, then, by prop.2.1.9, $\mathcal{A}$ is a semi-flower automaton.

Let us prove that $\mathcal{A}$ has at most one bpi. If $\mathcal{A}$ has exactly one bpi then, by theorem 2.3.12, it will follow $\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$.

-If there are no bpi's then either $H \cap K = \{\varepsilon\}$ or, by theorem 2.2.6, $H \cap K$ is a cyclic submonoid. So $rk(H \cap K)$ is at most two.

- Let us suppose now that there is at least one bpi, we will prove that it is the only one.

By contradiction, let x and x' be two different bpi's of $\mathcal{A}$. Let e_1 and e_2 be the edges ending at x. Let us consider the simple paths p and q from 1 to $i(e_1)$ and from 1 to $i(e_2)$, respectively. Let r be the simple path from x to 1. The paths pe_1r and qe_2r are cycles simple in 1 so, in particular, they are simple cycles.

Let p_1 be the longest prefix path in common between pe_1r and qe_2r, $pe_1r = p_1p_2$ and $qe_2r = p_1q_2$. One has that p_2 and q_2 are not null paths since pe_1r and qe_2r are cycles that are simple in 1. So $f(p_1)$ is a bpo. Let us call $\alpha = pe_1r$, $\beta = qe_2r$ and e_α (resp. e_β) the first edge of p_2 (resp. q_2).

Let us consider now the bpi x'. Let e_3 and e_4 be the edges ending at x'. Analogously as before let $p'e_3r'$ and $q'e_4r'$ be two cycles that are simple in 1. Let us call $\gamma = p'e_3r'$.

Let $\widetilde{\gamma}$ be the longest between the longest prefixes in common between α, γ and β, γ. Since α, β and γ are cycles that are simple in 1 then such prefixes are proper. Let us suppose that $\alpha = \widetilde{\gamma}\alpha'$ and $\gamma = \widetilde{\gamma}\gamma'$. In particular $f(\widetilde{\gamma})$ is a bpo. Since, by prop. 2.4.2 the product automaton $\mathcal{A}$ has at most one bpo, then $f(\widetilde{\gamma}) = f(p_1)$. Since α and γ are simple cycles then $\widetilde{\gamma} = p_1$.

Since, by lemma 2.4.2, every vertex has at most two edges going out from it then there are only e_α and e_β going out from $f(p_1) = f(\widetilde{\gamma})$. It follows that either $\gamma = \widetilde{\gamma}e_\alpha\gamma''$ or $\gamma = \widetilde{\gamma}e_\beta\gamma''$. In the first case γ and α have a prefix in common longer than $\widetilde{\gamma}$ contradiction! In the second case γ and β have a prefix in common longer than that one between γ and α contradiction! It is proved analogously if $\widetilde{\gamma}$ is the prefix path in common between β and γ.

So x is the unique bpi in $\mathcal{A}$ and applying theorem 2.3.12, it follows that

$$\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$$

(2)Let $rk(H \cap K) = \infty$

By prop. 2.1.8, $\mathcal{A}$ is not a semi-flower automaton and so there is a cycle c not visiting 1. Let c be a cycle in x. We may assume that c is simple otherwise we take as c the first simple cycle in which c is decomposed. We denote by v the label of c.

Let p be a simple path from 1 to x, u its label. Let q be a simple path from x to 1, w its label. So the cycle

$$pcq : 1 \xrightarrow{u} x \xrightarrow{v} x \xrightarrow{w} 1$$

is simple in 1. Moreover, for each $n > 0$, the cycle pc^nq is simple in 1.

Let p_1 be the longest prefix path in common between c and q, $c = p_1c_1$ and $q = p_1q_1$. The paths c_1 and q_1 are not null paths so $i(c_1) = i(q_1) = x_1$ is a bpo. Let v_1 be the label of p_1, v_2 the label of c_1 and w_1 the label of q_1.

We get

$$pp_1 : 1 \xrightarrow{uv_1} x_1, \quad c_1p_1 : x_1 \xrightarrow{v_2v_1} x_1, \quad q_1 : x_1 \xrightarrow{w_1} 1$$

with x_1 bpo.

Let $\alpha = pp_1$, $\beta = c_1p_1$ and $\gamma = q_1$. If α (resp γ) is not a simple path then let us take as α (resp. γ) the simple path from 1 to $f(\alpha)$ (resp. from 1 to $f(\gamma)$). If β is not a simple cycle then we will take as β the simple cycle from $i(\beta)$ to $f(\beta)$. So we have that

$$\alpha\beta\gamma : 1 \xrightarrow{uv_1} x_1 \xrightarrow{v_2u_1} x_1 \xrightarrow{w_1} 1$$

We have found that $C_{\mathcal{A}}{}' = \{\alpha(\beta)^n\gamma \mid n \geq 0\} \subseteq C_{\mathcal{A}}$ the set of cycles that are simple in 1.

We will prove now that these are the only cycles in $\mathcal{A}$ that are simple in 1. If there exists a cycle d that is simple in 1 and such that $d \notin C_{\mathcal{A}}{}'$, then, for each $c_i = \alpha(\beta)^i\gamma \in C_{\mathcal{A}}{}'$, let p_i be the longest prefix path in common between d and c_i. Let $P = \{p_i \mid i \geq 0\}$. Since $|p_i| < |d|$, for each i, we may consider $p_j \in P$ such that $|p_j| = max_{i\geq 0}\{|p_i|\}$. Then $d = p_jd'$ and $c_j = p_jc'$ and $f(p_j)$, the end vertex of p_j, is a bpo.

Since, by lemma 2.4.2, in $\mathcal{A}$ there is at most one bpi then $f(p_j) = f(\alpha)$.

Since p_j is a prefix of c_j and α, β and γ are simple then there exists $k \geq 0$ such that $p_j = \alpha\beta^k$. Let e_γ be the initial edge of γ and e_β be the initial edge of β. Since, by lemma 2.4.2, every vertex has at most two edges going out from it then there are only e_γ and e_β going out from $f(p_j)$. It follows that $d = \alpha\beta^k e_\gamma d'$ and $c_j = \alpha\beta^k e_\beta c'$ or $d = \alpha\beta^k e_\beta d'$ and $c_j = \alpha\beta^k e_\gamma c'$ for some $d', c' \in A^*$. In the first case c_k and d have a prefix in common longer than p_j contradiction! In the second case c_{k+1} and d have a prefix in common longer than p_j contradiction!

So $C_{\mathcal{A}} = \{\alpha(\beta)^n\gamma \mid n \geq 0\}$ and, if u (resp. v and w) is the label of α (resp. of β and of γ), we get that

$$H \cap K = Y_{\mathcal{A}}{}^* = (u(v)^*w)^*$$

□

It follows trivially the same result for H and K submonoids finitely generated by suffix sets of two elements.

2.5 The intersection of two submonoids: the non-prefix case

In the non-prefix case the techniques applied in the prefix case cannot be used anymore. This depends on the fact that, given a submonoid H generated by a non-prefix set, it does not exist a deterministic monoidal automaton recognizing H. Let us recall the following proposition of section 2.1.2:

Proposition 2.5.1 *Let $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ be a deterministic monoidal automaton. $\mathcal{A}$ recognizes a free submonoid generated by a prefix set.*

However, given a submonoid H generated by a finite non prefix set, one could consider ${\mathcal{A}_H}^D$, the deterministic automaton obtained from the flower automaton $\mathcal{A}_H$ of H applying the subset construction. This automaton has a number of final states greater than one and so it is not monoidal. We see an example in figure 2.23.

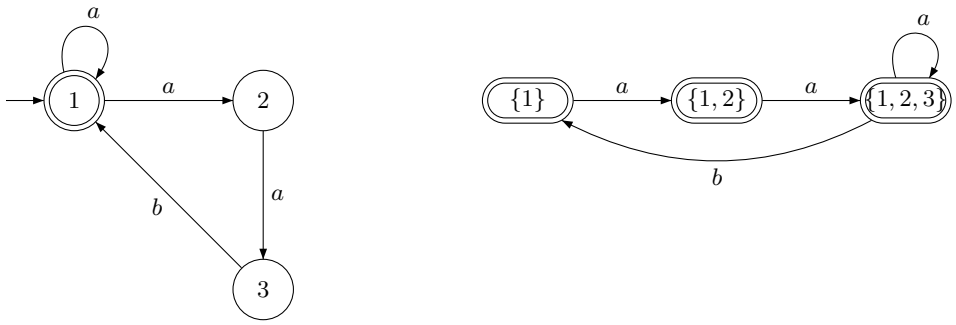

Figure 2.23: $\mathcal{A}_H$ flower automaton of $H = \{a, aab\}$ and ${\mathcal{A}_H}^D$ the automaton obtained from $\mathcal{A}_H$ by applying the subset construction.

If we consider H and K submonoids generated by finite non-prefix sets, then it is not useful to study the product automaton of ${\mathcal{A}_H}^D \times {\mathcal{A}_K}^D$. For instance neither one can easily understand if such automaton recognizes a finitely generated submonoid nor one can recognize the generators of the intersection's submonoid recognized to it.

Another possible approach to the study of the intersection of H and K, submonoids generated by finite non-prefix sets, is to consider the non deterministic product of the flower automata associated to H and K, $\mathcal{A}_H$ and $\mathcal{A}_K$. In this case it is not true the lemma 2.3.6 that links the sets of BPO in the product automaton with the sets of BPO of the starting automata.

Chapter 3

Automata and subgroups

The submonoids-monoidal automata correspondence we have talked about in chapter 2, takes inspiration from the well-known correspondence between finitely generated subgroups of a free group $F(A)$ and inverse automata on A. In this chapter we recall such correspondence and we underline some basic differences with the correspondence in the monoid's case.

To every inverse automaton $\mathcal{A}$ on A it is associated a graph $\Gamma_{\mathcal{A}}$, called *positive state graph of* $\mathcal{A}$. The main difference with the monoid's case is that, given an inverse automaton $\mathcal{A}$, there exists a formula linking the rank of the subgroup associated to it with the characteristics of the positive state graph of such automaton. In particular, if $\Gamma_{\mathcal{A}} = (Q, \mathcal{F})$ and H is the subgroup associated to $\mathcal{A}$ then

$$rk(H) = |\mathcal{F}| - |Q|$$

Such formula is essential for the proof of the Hanna Neumann conjecture true for positively generated subgroups. We present some examples pointing out the different behaviour between the case of free monoids and free groups.

Given an inverse automaton, to every spanning tree of its positive state graph it is associated a basis for the subgroup associated to it. It is well-known that such a basis is a Nielsen reduced basis. But it is not necessarily a strongly Nielsen basis. We propose two algorithms for finding a spanning tree of the positive state graph of an inverse automaton whose corresponding basis is a strongly Nielsen basis. An application of the second algorithm is an algorithm for the strongly Nielsen reduction.

In section 3.1 we recall the definition of the correspondence between inverse automata on A and subgroups of $F(A)$. A special kind of monoidal automata, that are the 'injective monoidal automata', can be considered as positive state graphs of inverse automata. So, in section 3.1.3, we discuss the differences between the ranks of submonoids and subgroups associated to such automata.

In section 3.2 we recall the arguments that lead to the Hanna Neumann inequality. We underline the difference with the free monoid's case presenting several examples.

In section 3.3 we recall definitions and properties of Nielsen bases and strongly Nielsen bases. In section 3.4.1 we give a first algorithm that constructs, starting from a spanning tree of the positive state graph of an inverse automaton, a spanning tree whose associated basis is a strongly Nielsen reduced basis.

In section 3.4.2 we prove that the breadth-first tree of the positive state graph of an inverse automaton is a spanning tree whose associated basis is a strongly Nielsen reduced basis. As an application of this result, in section 3.4.3, we give an algorithm for the strongly Nielsen reduction and we discuss the complexity of it.

3.1 The subgroups-automata correspondence

In this section we define the well-known correspondence between finitely generated subgroups of a free group on an alphabet A and 'inverse automata'. For the basic definitions and results we refer to [4] and [31].

This correspondence comes in a wider correspondence between subgroups of free groups and $A-$labelled graphs (cf. [32], [22]). We will mention it in section 3.2, where we discuss an approach to the Hanna Neumann conjecture on the intersection of two subgroups.

3.1.1 Automata recognizing subgroups

Let $\mathcal{A} = (Q, i, T, \delta)$ be a deterministic finite state automaton on A, with Q the set of states, i the initial state, T the set of final states and $\delta : Q \times A \longrightarrow Q$ the transition function.

If each $a \in A$ induces an injective function on Q and if $|T| = 1$ then $\mathcal{A}$ is an *injective* automaton. We will denote by t such unique final state. So, $\mathcal{A}$ is an injective automaton if, for each $a \in A$, for each $x \in Q$, there exists at most one edge ending at x with label a. Moreover, this is equivalent to the condition that the reverse of $\mathcal{A}$ (the automaton obtained by reversing all the arrows of $\mathcal{A}$) is deterministic. A trivial consequence of this fact is the following proposition:

Proposition 3.1.1 *Let $\mathcal{A} = (Q, i, t, \delta)$ be an injective automaton and let $x \in Q$.*
1)For each $w \in A^*$ there exists at most one path with label w ending at x.
2)For each $w \in A^*$ there exists at most one path with label w starting at x.

Proof. Let us prove the first statement. Let us suppose, by contradiction, that there exist $w \in A^*$ and two different paths p, q ending at x with label w. The paths p and q are not proper suffixes paths each other since w is not a proper suffix of itself. So, if r is the greatest suffix path in common between p and q then there are two edges with the same label ending at $i(r)$, the initial state of r, that is a contradiction since $\mathcal{A}$ is injective.

The second statement follows from the determinism of $\mathcal{A}$. □

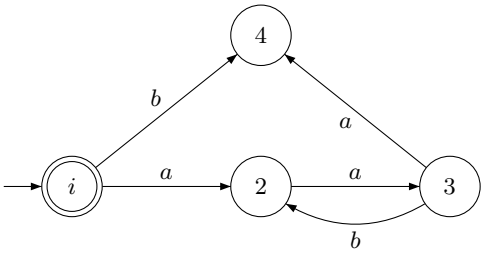

Figure 3.1: $\mathcal{A}$ injective automaton on A

In the example in figure 3.1 it is shown an injective automaton on A.

Let A^{-1} be a disjoint copy of A, together with a bijection $a \longrightarrow a^{-1}$ from A to A^{-1}. This bijection is extended to $(A \cup A^{-1})^*$ by letting $\varepsilon^{-1} = \varepsilon$, for each $a \in A$, $(a^{-1})^{-1} = a$ and, for each $n \geq 2$, $a_i \in A \cup A^{-1}$, $(a_1 \dots a_n)^{-1} = {a_n}^{-1} \dots {a_1}^{-1}$.

Definition 3.1.2 *An injective trim automaton $\mathcal{A} = (Q, i, t, \delta)$ on $A \cup A^{-1}$ is an inverse automaton if, for each $x, y \in Q$, for each $a \in A \cup A^{-1}$, one has*

$$\delta(x, a) = y \quad \textit{if and only if} \quad \delta(y, a^{-1}) = x$$

As seen in section 1.4, given $\mathcal{A} = (Q, i, i, \delta)$ inverse automaton, we can consider $\mathcal{F}$ the set of edges of $\mathcal{A}$ given by $\mathcal{F} = \{(x, a, y) \mid y = \delta(x, a)\}$. We have denoted the edge $(x, a, y) \in \mathcal{F}$ by $x \xrightarrow{a} y \in \mathcal{F}$. We will use indifferently either the notation δ or the notation $\mathcal{F}$.
So, in an inverse automaton

$$x \xrightarrow{a} y \quad \textit{if and only if} \quad y \xrightarrow{a^{-1}} x$$

From now on, we will consider inverse automata $\mathcal{A} = (Q, i, t, \mathcal{F})$ whose final state is equal to the initial one, $t = i$ and with $\mathcal{F} \neq \emptyset$.

Given an inverse automaton $\mathcal{A} = (Q, i, i, \mathcal{F})$ and an edge $e : x \xrightarrow{a} y$, with $x, y \in Q$, $a \in A \cup A^{-1}$, we denote by $e^{-1} : y \xrightarrow{a^{-1}} x$ the reverse edge of e. Given a path $p = e_1 \dots e_n$ in $\mathcal{A}$. We denote by p^{-1} the reverse path of p, $p^{-1} = {e_n}^{-1} \dots {e_1}^{-1}$. Given $P = \{p_1, \dots, p_n\}$ a finite set of paths in $\mathcal{A}$, we denote by P^{-1} the following set $P^{-1} = \{{p_1}^{-1}, \dots, {p_n}^{-1}\}$ and by $P^{\pm} = P \cup P^{-1}$.
Let us, moreover, denote by $\ell(p)$ the label of the path p and, given a finite set of paths P, by $\ell(P)$ the set of labels of the paths in P.

When representing an inverse automaton $\mathcal{A}$ on $A \cup A^{-1}$, it is convenient to represent the $A-$labelled edges, since the $A^{-1}-$labelled can be deduced immediately from them. This representation is called the *positive state graph* of $\mathcal{A}$. We denote it by $\Gamma_{\mathcal{A}}$.

In general, if $\mathcal{F}$ is the set of edges of $\mathcal{A}$, let us denote by $\mathcal{E}$ the set of edges of $\mathcal{F}$ with positive labels, that is $\mathcal{E} = \{e \in \mathcal{F} \mid \ell(e) \in A^*\}$. One has that $\mathcal{F} = \mathcal{E} \cup \mathcal{E}^{-1}$ and, in particular, $\Gamma_{\mathcal{A}} = (Q, i, i, \mathcal{E})$.

Remark 3.1.3 *Let us note that $\Gamma_{\mathcal{A}}$ is not necessarily a trim automaton as it is shown in the following example.*

Example 3.1.4 *In the example in figure 3.2 we see an inverse automaton $\mathcal{A}$ and its positive state graph $\Gamma_{\mathcal{A}}$. Let us note that $\Gamma_{\mathcal{A}}$ is not a trim automaton.*

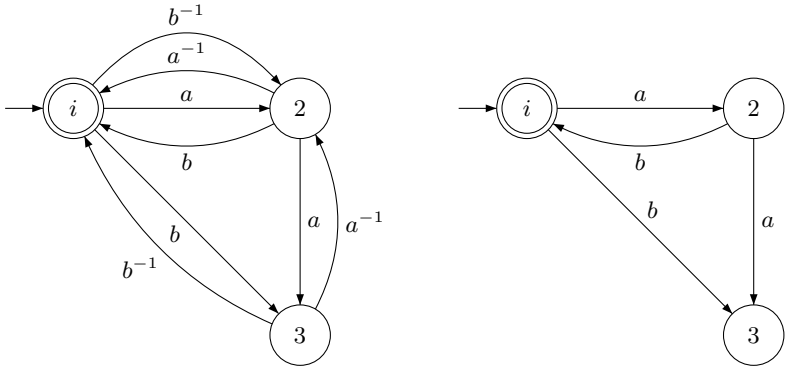

Figure 3.2: $\mathcal{A}$ inverse automaton and its positive state graph $\Gamma_{\mathcal{A}}$

There is a canonical way to *inversify* an injective automaton on A, $\mathcal{A} = (Q, i, t, \delta)$ such that $t = i$. Indeed, there is a unique way to extend δ to $Q \times (A \cup A^{-1})$ to make an inverse automaton on $A \cup A^{-1}$, by letting, for each $x, y \in Q$ and each $a \in A$, $\delta(y, a^{-1}) = x$ if and only if $\delta(x, a) = y$. Such an inverse automaton is denoted by A^{inv}. Let us note that A^{inv} has as positive state graph the automaton $\mathcal{A}$.

Example 3.1.5 *In the example in figure 3.3 we see an automaton $\mathcal{A}$ on A and $\mathcal{A}^{inv}$ the automaton obtained inversifing $\mathcal{A}$*

Let us note that, if $\mathcal{A} = (Q, i, i, \mathcal{F})$ is an injective automaton then the positive state graph of $\mathcal{A}^{inv}$ is $\mathcal{A}$.

We say that an inverse automaton is a *reduced inverse automaton* if every state $x \neq i$ is visited along a path from i to i labelled by a reduced word. This is equivalent to the condition that the graph representing the automaton does not have vertices of degree 1 (i.e. with an unique incident edge), except eventually the initial-final vertex.

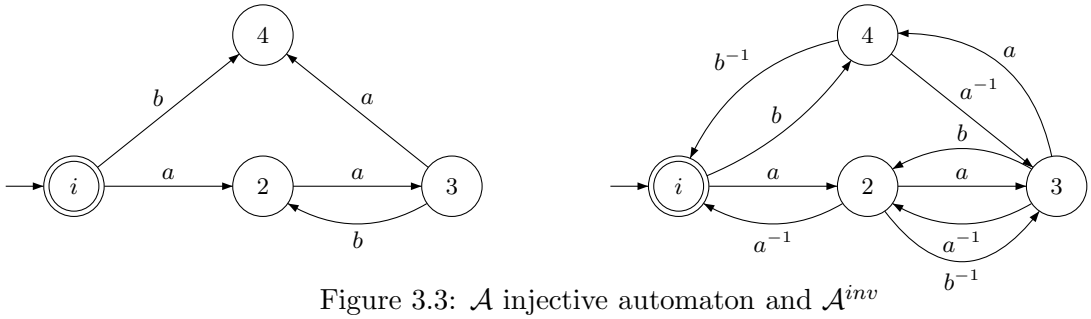

Figure 3.3: $\mathcal{A}$ injective automaton and $\mathcal{A}^{inv}$

An inverse automaton recognizes a language whose reduced words constitute a subgroup of $F(A)$ (cf. [31]). Given a word w in $(A \cup A^{-1})^*$ we indicate by $red(w)$ the reduced word associated to w.

Proposition 3.1.6 *Let $\mathcal{A} = (Q, i, i, \delta)$ be an inverse automaton. The language recognized by $\mathcal{A}$, $L(\mathcal{A})$, is a submonoid of $(A \cup A^{-1})^*$. The set $red(L(\mathcal{A}))$ is a subgroup of $F(A)$.*

We say that $red(L(\mathcal{A}))$ is the subgroup associated to $\mathcal{A}$. In the next subsection we will show that such subgroup is finitely generated and we will give an algorithm for finding a basis of it.

Example 3.1.7 *The example in figure 3.2 is a reduced inverse automaton $\mathcal{A}$ whose associated subgroup is $red(L(\mathcal{A})) =< aab^{-1}, ab >$. In figure 3.3 we have $\mathcal{A}^{inv}$ reduced inverse automaton whose associated subgroup is $red(L(\mathcal{A})) =< aaab^{-1}, aaba^{-1} >$.*

Given a subgroup H, there is at most a unique inverse reduced automaton whose associated subgroup is H, as it is shown in the next proposition and we denote it by $\mathcal{A}_H$ (cf. [31]). We moreover say that $\mathcal{A}_H$ *represents* H.

Proposition 3.1.8 *If $\mathcal{A}$ and $\mathcal{B}$ are reduced inverse automata such that $red(L(\mathcal{A})) = red(L(\mathcal{B}))$, then $\mathcal{A}$ and $\mathcal{B}$ are isomorphic (as graphs).*

In the next subsection we construct $\mathcal{A}_H$ when H is finitely generated.

Let now $\mathcal{A}$ be an inverse automaton. We say that a path in $\mathcal{A}$ is a *reduced path* if it does not contain as subpaths ee^{-1} and $e^{-1}e$, for each e edge of $\mathcal{A}$. For each p path in $\mathcal{A}$, we define the *reduced path associated to* p as the path obtained from p by removing iteratively, if there are, the subpaths ee^{-1} and $e^{-1}e$, with e edge. It is denoted by $red(p)$. Let us indicate by $|p|$ the length (i.e. the number of edges) of $red(p)$. If p is a path in a graph (or in a multi-graph) we denote by (p) its length. Let us remember that, for each $u \in (A \cup A^{-1})^*$ we indicate by $|u|$ the length of $red(u)$, the reduced word associated to u and by (u) the length of the word u.

We have the following proposition that links the length of the reduced word associated to a path with the length of the correspondent reduced path.

Proposition 3.1.9 *Let $\mathcal{A} = (Q, i, i, \mathcal{F})$ be an inverse automaton. If p is a path in $\mathcal{A}$ with label u then $|p| = |u|$.*

Proof. Let p be a path in $\mathcal{A}$ with label u, $p = e_1 \dots e_n$ with $e_i \in \mathcal{F}$, for each i, and $u = a_1 \dots a_n$ with $a_i \in A \cup A^{-1}$ such that $\ell(e_i) = a_i$. Let us prove that, for each $i = 1, \dots, n$, $e_i e_i^{-1}$ (resp. $e_i^{-1} e_i$) is in p if and only if $a_i {a_i}^{-1}$ (resp. $a_i^{-1} a_i$) is in u.

If there exists i such that $e_{i+1} = {e_i}^{-1}$ then $a_{i+1} = {a_i}^{-1}$. Conversely if there exists i such that $a_{i+1} = {a_i}^{-1}$ then $\ell(e_i) = a_i$ and $\ell(e_{i+1}) = a_i^{-1}$ and since, by the injectivity of $\mathcal{A}$, the unique edge starting at $f(e_i)$ with label ${a_i}^{-1}$ is ${e_i}^{-1}$ then $e_{i+1} = {e_i}^{-1}$. □

Let $\mathcal{A} = (Q, i, i, \mathcal{E} \cup \mathcal{E}^{-1})$ be an inverse automaton and $\Gamma_{\mathcal{A}}$ its positive state graph. Let $\Gamma'_{\mathcal{A}}$ be the undirected multigraph associated to $\Gamma_{\mathcal{A}}$. It is defined as $\Gamma'_{\mathcal{A}} = (Q, \overline{\mathcal{E}}, f)$ where $\overline{\mathcal{E}} = \{\overline{e} \mid e \in \mathcal{E}\}$ is a set in bijection with $\mathcal{E}$ and

$$f : \overline{\mathcal{E}} \longrightarrow \{\{x, a, y\} \mid x, y \in Q, a \in A\}$$

is such that $f(\overline{e}) = \{i(e), \ell(e), f(e)\}$.

Let $T = (Q, \mathcal{F}(T))$ be a spanning tree of $\Gamma_{\mathcal{A}}$. In particular, it is a subgraph of $\Gamma'_{\mathcal{A}}$ and $\mathcal{F}(T) \subseteq \overline{\mathcal{E}}$.

Let $\widetilde{T}$ be the directed graph associated to T, that is $\widetilde{T} = (Q, \mathcal{F}(\widetilde{T}))$ with

$$\mathcal{F}(\widetilde{T}) = \{e \in \mathcal{E} \mid \overline{e} \in \mathcal{F}(T)\}$$

Let us denote by $\widetilde{T}^{inv}$ the inverse automaton obtained inversifing the directed subgraph $\widetilde{T}$.

Let us consider now such inverse automaton $\widetilde{T}^{inv}$. Its positive state graph $\widetilde{T}$ is in particular a graph and T is the undirected graph associated to $\widetilde{T}$. Since T is a labelled undirected graph then we can consider

$$\mathcal{F}(T) = \{\{x, a, y\} \mid x, y \in Q,\ a \in A\}$$

And we have that if $\{x, a, y\} \in \mathcal{F}(T)$ then either $(x, a, y) \in \mathcal{F}(\widetilde{T})$ or $(y, a, x) \in \mathcal{F}(\widetilde{T})$.

In the following, starting from a path p in T we construct a path $\widetilde{p}$ in $\widetilde{T}^{inv}$ and, conversely, starting from a path p in $\widetilde{T}^{inv}$ we construct a path $\overline{p}$ in T. We will prove, in prop. 3.1.12, that such functions are invertible, that is, for each p path in T, $\overline{\widetilde{p}}$ and for each q path in $\widetilde{T}^{inv}$, $\widetilde{\overline{q}}$.

In particular if p is a path in T then if we go along one of its edge in $\widetilde{T}$ in arrow's direction then such edge will be the same in $\widetilde{p}$ otherwise it will be the edge reverse of

it. Conversely if p is a path in $\widetilde{T}^{inv}$ then the path $\overline{p}$ will be the sequence of its vertices. More formally, let us define now such functions:

$$^{\sim}: T \longrightarrow \widetilde{T}^{inv}$$

Let $p = (x_1, \ldots, x_{n+1})$ be a path in T. In particular $\{x_i, a, x_{i+1}\} \in \mathcal{F}(T)$, for each $i = 1, \ldots, n$, and either $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ or $(x_{i+1}, a, x_i) \in \mathcal{F}(\widetilde{T})$.

Then we can consider a path $\widetilde{p} = g_1 \ldots g_n$ in $\widetilde{T}^{inv}$ such that $g_i = (x_i, a, x_{i+1})$ if $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ and $g_i = (x_i, a^{-1}, x_{i+1})$ if $(x_{i+1}, a, x_i) \in \mathcal{F}(\widetilde{T})$.

$$- : \widetilde{T}^{inv} \longrightarrow T$$

Let $p = g_1 \ldots g_n$ be a path in $\widetilde{T}^{inv}$, with $g_i = (x_i, a, x_{i+1}) \in (\mathcal{F}(\widetilde{T}) \cup \mathcal{F}(\widetilde{T})^{-1})$. Then we can associate to it the path $\overline{p} = (x_1, \ldots, x_{n+1})$ in T. This is a path since if $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ then $\{x_i, a, x_{i+1}\} \in \mathcal{F}(T)$ and if $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})^{-1}$ then $\{x_i, a^{-1}, x_{i+1}\} \in \mathcal{F}(T)$.

Remark 3.1.10 *Let us note that if $p = g_1 \ldots g_n$, with $g_i = (x_i, a, x_{i+1}) \in (\mathcal{E} \cup \mathcal{E}^{-1})$, is a path in $\mathcal{A}$, then we can construct a path $\overline{p}'$ in $\Gamma'_{\mathcal{A}}$ given by the sequence of edges $\overline{p}' = h_1 \ldots h_n$, where $h_i = \overline{g_i}$, if $h_i \in \mathcal{E}$, and $h_i = \overline{g_i^{-1}}$, if $h_i \in \mathcal{E}^{-1}$.*

Such path has the same sequence of vertices of p. Moreover if $p = g_1 \ldots g_n$ is in $\widetilde{T}^{inv}$ then $\overline{p}' = \overline{p}$.

Example 3.1.11 *In the example in figure 3.4 we see an inverse automaton $\mathcal{A}$ and its positive state graph $\Gamma_{\mathcal{A}}$. The edges underlined in $\Gamma_{\mathcal{A}}$ are those ones of T spanning tree of $\Gamma_{\mathcal{A}}$. The edges underlined in $\mathcal{A}$ are those ones of $\widetilde{T}^{inv}$. Let us consider the path $p = (i, 2, 3)$ in T. Then $\widetilde{p} : i \xrightarrow{a} 2 \xrightarrow{b^{-1}} 3$. Moreover one has that $\overline{\widetilde{p}} = (i, 2, 3) = p$.*

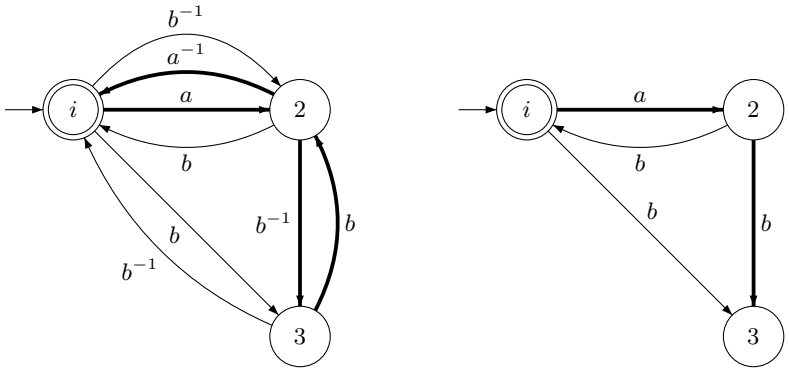

Figure 3.4: $\mathcal{A}$ inverse automaton and its positive state graph $\Gamma_{\mathcal{A}}$

Let us prove now that the functions above defined are invertible:

Proposition 3.1.12 *For each p path in T, $\overline{\widetilde{p}} = p$ and, for each p path in $\widetilde{T}^{inv}$, $\widetilde{\overline{p}} = p$.*

Proof.
1)$\overline{\widetilde{p}} = p$
Let p be a path in T, then $p = (x_1, \ldots, x_{n+1})$. It follows easily that $\overline{\widetilde{p}} = p$.
2)$\widetilde{\overline{p}} = p$
Let p be a path in $\widetilde{T}^{inv}$, then $p = g_1 \ldots g_n$. Let $g_i = (x_i, a, x_{i+1})$ with $a \in A \cup A^{-1}$. One has that $\overline{p} = (x_1, \ldots, x_{n+1})$ with $\{x_i, a, x_{i+1}\} \in \mathcal{F}(T)$ if $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ and $\{x_i, a^{-1}, x_{i+1}\} \in \mathcal{F}(T)$ if $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})^{-1}$.

One has that $\widetilde{\overline{p}} = f_1 \ldots f_n$ with $f_i = (x_i, a, x_{i+1})$ if $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ and $f_i = (x_i, a^{-1}, x_{i+1})$ if $(x_{i+1}, a, x_i) \in \mathcal{F}(\widetilde{T})$.

Let $g_i = (x_i, a, x_{i+1})$. If $a \in A$ then $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$. So $f_i = (x_i, a, x_{i+1}) = g_i$.

If $a \in A^{-1}$ then $g_i = (x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})^{-1}$ and $(x_{i+1}, a^{-1}, x_i) \in \mathcal{F}(\widetilde{T})$. So $f_i = (x_i, a, x_{i+1}) = g_i$. □

Let us remember that, given T a spanning tree of $\Gamma_{\mathcal{A}}$ and p a path in $\Gamma_{\mathcal{A}}$, we will say that p *is in* T if all the edges of p are edges of T. We moreover say that an edge e *occurs in* p if e is an edge of p.

Given T a spanning tree of $\Gamma_{\mathcal{A}}$ and p a path in $\mathcal{A}$ we say that p *is contained in* T if p is in $\widetilde{T}^{inv}$. We will say that an edge e *occurs in* p if either e or e^{-1} occurs in p (i.e. $\overline{e}$ occurs in $\overline{p}$).

However the different nomenclature will be understandable from the context.

3.1.2 The subgroups-inverse automata correspondence

Let H be a finitely generated subgroup of $F(A)$. Let H be generated by the finite set Y of $F(A)$, $H =< Y >$. We construct a positive state graph of the reduced inverse automaton representing H, $\mathcal{A}_H$, in three steps.

$$\mathbf{H} =< \mathbf{Y} > \longrightarrow \mathcal{A}_{\mathbf{H}}$$

1)We construct a set of $|Y|$ cycles around a common distinguished state i, each labelled by an element of Y. Since only the $A-$labelled edges are indicated, an inverse letter a^{-1} in a word of Y gives rise to an a-labelled edge in the reverse direction on the corresponding cycle.

2)We iteratively identify identically-labelled pairs of edges starting or ending at the same state.

3)We iteratively remove states of degree 1 other than i.

Example 3.1.13 *In figure 3.5 there is the construction of the positive state graph of* $\mathcal{A}_{<Y>}$ *for* $Y = \{b^2a^{-1}b, a^3b^{-1}, b^{-2}a\}$.

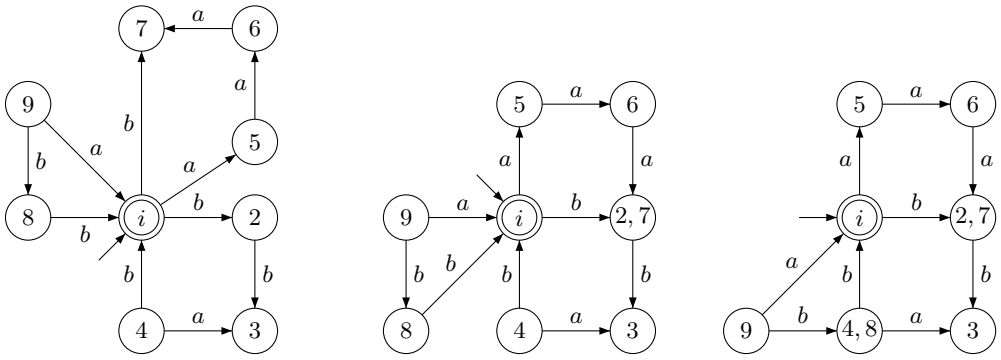

Figure 3.5:

In the previous subsection we have seen that to a reduced inverse automaton $\mathcal{A}$ it is associated the subgroup $red(L(\mathcal{A}))$. Let $\mathcal{A} = (Q, i, i, \mathcal{F})$ be a reduced inverse automaton. Let us see how to construct a finite set of words Y such that $\mathcal{A} = \mathcal{A}_{<Y>}$, and Y is a basis of $red(L(\mathcal{A}))$.

$$\mathcal{A} \longrightarrow \mathbf{Y}, \quad \mathbf{with} \ < \mathbf{Y} >= \mathbf{red}(\mathbf{L}(\mathcal{A}))$$

Let $T = (Q, \mathcal{F}(T))$ be a spanning tree of $\Gamma_{\mathcal{A}}$, $\widetilde{T}$ be the directed graph associated to T and let us consider $\widetilde{T}^{inv}$, the subgraph of $\mathcal{A}$ consisting of all the edges of $\widetilde{T}$ and their inverses. For each $x \in Q$, there exists a unique reduced path from i to x in $\widetilde{T}^{inv}$, as it is shown in the following proposition:

Proposition 3.1.14 *Let* $\mathcal{A} = (Q, i, i, \mathcal{E} \cup \mathcal{E}^{-1})$ *be an inverse automaton,* $\Gamma_{\mathcal{A}}$ *its positive state graph and* $T = (Q, \mathcal{F}(T))$ *a spanning tree of* $\Gamma_{\mathcal{A}}$. *Then, for each* $x \in Q$, *there exists a unique reduced path from* i *to* x *contained in* T.

Proof. Let $\mathcal{A} = (Q, i, i, \mathcal{E} \cup \mathcal{E}^{-1})$ be an inverse automaton, $\Gamma_{\mathcal{A}}$ its positive state graph, $T = (Q, \mathcal{F}(T))$ a spanning tree of $\Gamma_{\mathcal{A}}$ and $\widetilde{T} = (Q, \mathcal{F}(\widetilde{T}))$ the directed graph associated to it.

Let $x \in Q$. It is well known that there exists a unique simple path from i to x in T, let it be $p = (x_1, \ldots, x_{n+1})$. One has that, for each $i = 1, \ldots, n$, $\{x_i, a, x_{i+1}\} \in \mathcal{F}(T)$. Let us consider $\widetilde{p} = g_1 \ldots g_n$ in $\widetilde{T}^{inv}$, as defined in section 3.1.1, such that $g_i = (x_i, a, x_{i+1})$ if $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ and $g_i = (x_i, a^{-1}, x_{i+1})$ if $(x_{i+1}, a, x_i) \in \mathcal{F}(\widetilde{T})$.

Let us prove, in the following two points, that $\widetilde{p}$ is a reduced path and moreover it is the unique reduced path contained in T.

-$\widetilde{p}$ is a reduced path contained in T.

If $\widetilde{p}$ is not reduced, then $\widetilde{p}$ contains ee^{-1}, for some edge e. The sequence of vertices in $\widetilde{p}$ is the same as the sequence of p, so p contains twice the same vertex $i(e)$ and this is a contradiction since p is a simple path.

-$\widetilde{p}$ is the unique reduced path contained in T.

We will now prove that if there exists another reduced path $q = g_1 \dots g_n$ from i to x contained in T, $q \neq \widetilde{p}$, then $\overline{q}$ is a simple path in T. To prove this fact we will show that $\overline{q}$ does not contain any edge twice consecutively. Let $\overline{q} = (x_1, \dots, x_{n+1})$ with, for each $i = 1, \dots, n$, $\{x_i, a, x_{i+1}\} \in \mathcal{F}(T)$. If, by contradiction, there exists $i \in \{1, \dots, n\}$ such that $\{x_i, a, x_{i+1}\} = \{x_{i+1}, a, x_{i+2}\}$ then $x_i = x_{i+2}$. Since T is a tree then $x_i \neq x_{i+1}$.

Since T is the undirected subgraph associated to $\widetilde{T}$ then either $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ or $(x_{i+1}, a, x_i) \in \mathcal{F}(\widetilde{T})$. Let us suppose that $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ and $g_i = (x_i, a, x_{i+1})$.

So $(x_i, a, x_{i+1}) = (x_{i+2}, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})$ and one has that $g_{i+1} = (x_{i+1}, a^{-1}, x_i) = {g_i}^{-1}$. So $g_i {g_i}^{-1}$ is contained in $\widetilde{\overline{q}} = q$ by prop.3.1.12, that is a contradiction since q is reduced. If $(x_i, a, x_{i+1}) \in \mathcal{F}(\widetilde{T})^{-1}$ and $g_i = (x_i, a, x_{i+1})$ then, as before, $g_{i+1} = {g_i}^{-1} = (x_{i+1}, a^{-1}, x_i) \in \mathcal{F}(\widetilde{T})$ that is a contradiction.

By prop. 1.4.6 $\overline{q}$ is a simple path which is a contradiction, since there exists a unique simple path from i to x in T and $p \neq \overline{q}$. □

Let us so denote by p_x the unique reduced path from i to x in $\widetilde{T}^{inv}$. The following proposition (cf. [4]) associates to T a finite basis for $red(L(\mathcal{A}))$:

Proposition 3.1.15 *Let $\mathcal{A} = (Q, i, i, \mathcal{E} \cup \mathcal{E}^{-1})$ be a reduced inverse automaton and $\Gamma_{\mathcal{A}}$ its positive state graph. Let T be a spanning tree of $\Gamma_{\mathcal{A}}$ and $\widetilde{T} = (Q, \mathcal{F}(\widetilde{T}))$ the directed graph associated to it. Let, for each $e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})$,*

$$b_e = p_{i(e)} e p_{f(e)}^{-1}$$

and

$$B_T = \{b_e \ , \quad \forall e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})\}$$

Then $Y = \ell(B_T)$ is a basis for $red(L(\mathcal{A}))$.

The cardinality of such basis is linked with the number of vertices and of edges of $\Gamma_{\mathcal{A}}$, as it is stated in the following proposition (cf. [4]):

Proposition 3.1.16 *Let $\mathcal{A}$ be a reduced inverse automaton, $\Gamma_{\mathcal{A}} = (Q, \mathcal{F})$ its positive state graph and $T = (Q, \mathcal{F}(T))$ a spanning tree of $\Gamma_{\mathcal{A}}$. Then*

$$|\ell(B_T)| = |\mathcal{F}| - |Q| + 1$$

Proof. Since $|\ell(B_T)| = |\mathcal{F}| - |\mathcal{F}(T)|$ and since $|\mathcal{F}(T)| = |Q| - 1$, being T a spanning tree of $\Gamma_{\mathcal{A}}$, one has that $rk(red(L(\mathcal{A}))) = |\ell(B_T)| = |\mathcal{F}| - |Q| + 1$. □

Example 3.1.17 *In figure 3.6 we see the positive state graph of an inverse automaton $\mathcal{A}$. The edges underlined are those ones of T spanning tree in i. One has $\ell(B_T) = \{aba^{-1}a^{-1}, aaab^{-1}, ba, bb\}$. To $\mathcal{A}$ it is associated the subgroup $red(L(\mathcal{A})) =< \ell(B_T) >$.*

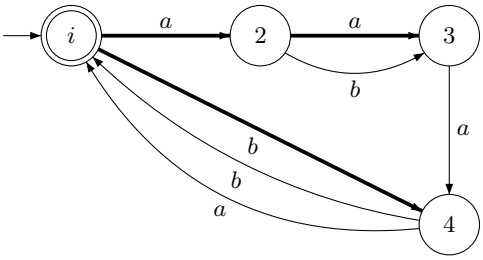

Figure 3.6: $\Gamma_{\mathcal{A}}$ positive state graph of an inverse automaton.

A useful property of paths contained in T is stated in the following proposition:

Proposition 3.1.18 *Let $\mathcal{A}$ be a reduced inverse automaton and T a spanning tree of $\Gamma_{\mathcal{A}}$. Let p be a reduced path contained in T. Then every edge in p occurs just once.*

Proof. Since p is a reduced path contained in T then $\overline{p}$ is in T. By the proof of proposition 3.1.14, $\overline{p}$ is a simple path so, by remark 1.4.8, every edge in $\overline{p}$ occurs just once. □

Let us make now two considerations on the paths b_e considered in $\mathcal{A}$

Proposition 3.1.19 *For each e not occurring in T, b_e is a reduced path.*

Proof. One has that $b_e = p_{i(e)} e p_{f(e)}^{-1}$. The paths $p_{i(e)}$ and $p_{f(e)}^{-1}$ are reduced, by definition. Since e is the unique edge in b_e not contained in T, the thesis follows. □

Let $\mathcal{A}$ be a reduced inverse automaton. For each $T = (Q, \mathcal{F}(T))$ spanning tree of $\Gamma_{\mathcal{A}} = (Q, \mathcal{F})$, we have seen that $\ell(B_T)$ is a basis for the subgroup associated to $\mathcal{A}$ and

$$|\ell(B_T)| = |\mathcal{F}| - |Q| + 1$$

In the class of monoidal automata we have seen (see theorem 2.2.10) that the same link holds when $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ is a semi-flower unambiguous automaton with a unique bpi, that is $rk(L(\mathcal{A})) = |\mathcal{F}| - |Q| + 1$.

In particular, taking T the BFS tree of $\mathcal{A}$ in 1, one can associate to every edge not in T a generator for $L(\mathcal{A})$ (see remark 2.2.13).

But if $\mathcal{A} = (Q, 1, 1, \mathcal{F})$ is a semi-flower automaton with more than one bpi then this formula does not hold in general, as it is shown in the following example.

Example 3.1.20 *In figure 3.7 it is represented a semi-flower automaton with more than one bpi, $\mathcal{A} = (Q, 1, 1, \mathcal{F})$. Moreover, it is the positive state graph of the inverse automaton $\mathcal{A}^{inv}$. The edges underlined are those ones of T spanning tree in 1. The automaton $\mathcal{A}$ recognizes the submonoid generated by*

$$Y_{\mathcal{A}} = \{aaab, aaaa, abab, abaa, bb, ba\}$$

One has

$$rk(L(\mathcal{A})) = 6 > |\mathcal{F}| - |Q| + 1 = 4$$

To the automaton $\mathcal{A}^{inv}$ it is associated the subgroup generated by

$$\ell(B_T) = \{aba^{-1}a^{-1}, aaab^{-1}, ba, bb\}$$

One has

$$rk(red(L(\mathcal{A}^{inv}))) = 4 = |\mathcal{F}| - |Q| + 1$$

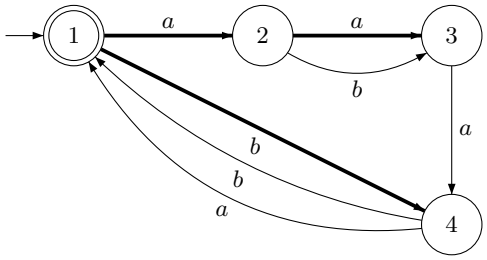

Figure 3.7: $\mathcal{A}$ semi-flower automaton with more than one bpi and positive state graph of the inverse automaton $\mathcal{A}^{inv}$.

Moreover, let us note that, for reduced inverse automata, to every spanning tree of the positive state graph it is associated a basis for the subgroup associated to it, while in case of semi-flower automata with a unique bpi just to a BFS tree it is associated a basis for the submonoid generated by it. This depends on the fact that the edges in the positive state graph of an inverse automaton can be read also in the opposite direction with inverse label.

3.1.3 Automata associated to subgroups and to submonoids: bases

Every injective monoidal automaton can be viewed as the positive state graph of an inverse automaton. In particular, if $\mathcal{A}$ is a injective monoidal automaton we have seen that we can inversify $\mathcal{A}$ and obtain the inverse automaton $\mathcal{A}^{inv}$ whose positive state graph is $\mathcal{A}$.

If $\mathcal{A}$ is a monoidal injective automaton, let us denote by $H = L(\mathcal{A})$ the submonoid recognized by $\mathcal{A}$ and by $\overline{H} = red(L(\mathcal{A}^{inv}))$ the subgroup associated to $\mathcal{A}^{inv}$.

In particular, while $Y_{\mathcal{A}}$ is the minimal set of generators of the submonoid $L(\mathcal{A})$, one has that to $\mathcal{A}^{inv}$ it is associated a subgroup whose bases can be determined by choosing spanning trees of $\mathcal{A}$.

As we have seen in the example in figure 3.7, in general such bases for the submonoid and for the subgroup associated, are different and moreover $L(\mathcal{A})$ and $red(L(\mathcal{A}^{inv}))$ have different rank.

Just in the case that $\mathcal{A}$ is an injective semi-flower automaton with a unique bpi we have that $rk(H) = rk(\overline{H})$. The following examples come in such case.

Example 3.1.21 *In figure 3.8 we have $\mathcal{A}$, semi-flower automaton with a unique bpi and the positive state graph of an inverse automaton $\mathcal{A}^{inv}$. The subgraph of $\mathcal{A}$ with edges underlined is the BFS tree T of $\mathcal{A}$ in* 1. *The automaton $\mathcal{A}$ recognizes the submonoid generated by*

$$Y_{\mathcal{A}} = \{aab, ab\}$$

To the automaton $\mathcal{A}^{inv}$ it is associated the subgroup generated by

$$\ell(B_T) = \{aab, ab\}$$

One has that

$$Y_{\mathcal{A}} = \ell(B_T)$$

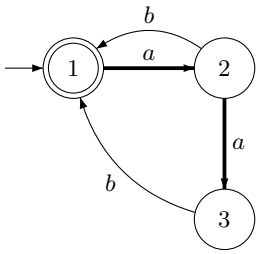

Figure 3.8: $\mathcal{A}$ semi-flower automaton with a unique bpi and the positive state graph of an inverse automaton $\mathcal{A}^{inv}$.

Example 3.1.22 *In figure 3.9 there is $\mathcal{A}$, semi-flower automaton with a unique bpi and the positive state graph of an inverse automaton $\mathcal{A}^{inv}$. The subgraph of $\mathcal{A}$ with edges underlined is the BFS tree T of $\mathcal{A}$ in* 1. *The automaton $\mathcal{A}$ recognizes the submonoid H generated by*

$$Y_{\mathcal{A}} = \{aab, bb\}$$

To the automaton $\mathcal{A}^{inv}$ it is associated the subgroup $\overline{H}$ generated by

$$\ell(B_T) = \{aab^{-1}, bb\}$$

One has that

$$rk(H) = rk(\overline{H})$$

and

$$Y_{\mathcal{A}} \neq \ell(B_T)$$

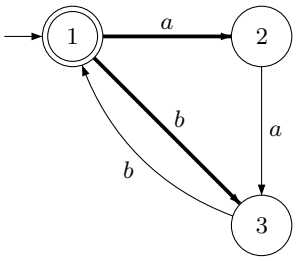

Figure 3.9: $\mathcal{A}$ semi-flower automaton with a unique bpi and the positive state graph of an inverse automaton $\mathcal{A}^{inv}$.

In general this does not happen: if $\mathcal{A}$ is an injective semi-flower automaton with more than one bpi then it is not more true that $rk(H) = rk(\overline{H})$ (see example in figure 3.7). Moreover if $\mathcal{A}$ is an injective monoidal not semi-flower automaton then $rk(H)$ can be infinite and $rk(\overline{H})$ finite (see the following example).

Example 3.1.23 *In figure 3.10 there is $\mathcal{A}$, monoidal automaton and the positive state graph of an inverse automaton $\mathcal{A}^{inv}$. The subgraph of $\mathcal{A}$ with edges underlined is the spanning tree T of $\mathcal{A}$. The automaton $\mathcal{A}$ recognizes the submonoid generated by*

$$Y_{\mathcal{A}} = \{ab^*ab\}$$

and to $\mathcal{A}^{inv}$ it is associated the subgroup generated by

$$\ell(B_T) = \{aba^{-1}, aab\}$$

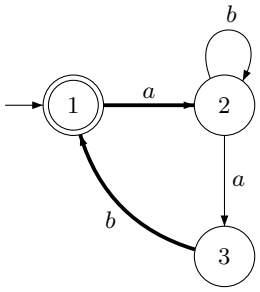

Figure 3.10: $\mathcal{A}$ monoidal automaton and the positive state graph of an inverse automaton $\mathcal{A}^{inv}$.

One has that

$$rk(H) = \infty$$

and

$$rk(\overline{H}) = 2$$

Remark 3.1.24 *The problem of searching the basis of the subgroup generated by a submonoid is related to the research, given a submonoid H, of the basis of the smallest submonoid biunitary containing H, where a submonoid H is biunitary if*

$$HH^{-1} = H^{-1}H = H$$

where

$$H^{-1}H = \{u \in A^* \mid vu \in H \text{ , } for \text{ } some \text{ } v \in H\}$$

and

$$HH^{-1} = \{u \in A^* \mid uv \in H \text{ , } for \text{ } some \text{ } v \in H\}$$

In general, given a submonoid H the basis of the smallest biunitary containing H is different from the basis of the subgroup generated by H. For example, if

$$X = \{ab, aaa, aaba, aabb, baa, bba, bbb\}$$

then the submonoid X^* is biunitary while the subgroup generated by X^* is

$$< X >= \{a, b\}$$

Resuming, given an injective monoidal automaton $\mathcal{A}$, it recognizes a submonoid that has as unique minimal set of generators $Y_{\mathcal{A}}$, the set of labels of the cycles that are simple in 1. On the contrary to $\mathcal{A}^{inv}$ it is associated a subgroup that can have more bases, associated to spanning trees, and in general $Y_{\mathcal{A}}$ is not a basis for such subgroup and the cardinality of $Y_{\mathcal{A}}$ is different from the rank of the subgroup associated to $\mathcal{A}^{inv}$.

In general, given an inverse automaton, since to every spanning tree of its positive state graph it corresponds a basis of the subgroup associated to it, one can pose the problem of choosing the spanning tree whose associated basis is a 'best possible'. In particular the bases where the cancellation of the elements in the product is reduced to the minimum: such bases are called 'strongly Nielsen reduced bases '. We will discuss these bases in section 3.3 and, in section 3.4, we will give two algorithms to choose the spanning tree for the positive state graph of an inverse automaton whose associated bases are such bases.

3.2 About the intersection of two subgroups: the Hanna Neumann conjecture

In this section we recall some properties of the intersection of two finitely generated subgroups of a free group. In particular, we are interested in an upper bound for the rank of the intersection of two finitely generated subgroups of a free group, and we describe the results so far obtained. Then we present some examples pointing out the different behaviour between the case of free monoids and free groups.

Let A be an alphabet and $F(A)$ the free group on A. Unlike the case of monoids, by the theorem of Nielsen-Schreier any subgroup of a free group is free.

We have already observed, in section 1.3, that a free group can have many bases and that all of them have the same cardinality. Hence the *rank* of a free group F is defined as the cardinality of a basis of F and the *reduced rank* of a free group F is defined as $\widetilde{rk}(F) = max(rk(F) - 1, 0)$.

It is known since Howson's 1954 paper ([13]) that the intersection of two finitely generated subgroups of a free group is finitely generated. In particular, Howson gave an upper bound for the rank of the intersection $H \cap K$ in terms of H and K.

In 1956 Hanna Neumann ([23]) improved Howson's bound by showing that if H and K are two subgroups of finite rank of a free group then $\widetilde{rk}(H \cap K) \leq 2\widetilde{rk}(H)\widetilde{rk}(K)$. Then she made the following conjecture, known nowadays as the

Hanna Neumann conjecture

$$\widetilde{rk}(H \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K)$$

In 1991 Walter Neumann ([24]) formulated a stronger conjecture known as 'Strengthened Hanna Neumann conjecture' (in short SHN).

Let H and K be finitely generated subgroups of a a free group $F(A)$ and let $H\backslash F(A)/K = \{HuK \mid u \in F(A)\}$ be the set of double $H-$cosets. If $u \in F(A)$, H^u denotes the conjugate $H^u = u^{-1}Hu$. Let us note that if $HuK = HvK$, then $\widetilde{rk}(H^u \cap K) = \widetilde{rk}(H^v \cap K)$.

Strengthened Hanna Neumann conjecture

$$\sum_{HgK \in H\backslash /K} \widetilde{rk}(H^g \cap K) \leq \widetilde{rk}(H)\widetilde{rk}(K) \tag{3.2.1}$$

If H is a finitely generated subgroup of a free group $F(A)$, we say that *SHN holds for* H if the inequality 3.2.1 holds for every finitely generated subgroup K of $F(A)$.

Walter Neumann ([24]) extended Burns' bound to the strengthened version of the conjecture and Tardos ([33]) proved that SHN holds for all subgroups of rank 2. Moreover, Dicks and Formanek in [8] proved in particular that SHN holds for subgroups of rank 3.

Finally, in 2002 Meakin and Weil ([22]) proved that SHN holds for the class of positively generated subgroups of the free group $F(A)$ on a finite alphabet A, that are generated by words on A^*. This last result suggested us to propose the problem of Hanna Neumann for finitely generated submonoids of a free monoid, in the case that their intersection is finitely generated.

Some of the basic tools in dealing with the Hanna Neumann conjecture for free groups make use of the representation of subgroups of the free group by graphs (or inverse automata).

In particular, in the paper of Meakin and Weil (cf. [22]) each subgroup of the free group $F(A)$ is represented by a *directed A-labeled graph* (that is, a directed graph whose edges are labeled by elements of A).

Let us give now some definitions in order to better understand the idea of the proof of Meakin and Weil.

In A-labeled directed graphs we consider paths which may traverse edges in either direction. The convention is that traversing an $a-$labeled edge in the backward direction carries the label a^{-1}. We say that such a graph is *admissible* if it is connected and, for each $a \in A$ and each vertex v, there is at most one $a-$labeled edge starting at v and at most one $a-$labeled edge ending at v.

Let us define $\Gamma(H)$. First, let $\Gamma_0(H)$ be the multigraph whose vertex set is the set of the right H-cosets, $V = \{Hx \mid x \in F(A)\}$, whose edge set is $E = \{Hx \mid x \in F(A)\} \times A$ and whose functions $i, f : E \longrightarrow V$ are such that, for each $(Hx, a) \in E$, $i(Hx, a) = Hx$ and $f(Hx, a) = Hxa$ (that is, the edge (Hx, a) starts at Hx and ends at Hxa). Then $\Gamma(H)$ is the *H-core* of $\Gamma_0(H)$, that is, the least subgraph of $\Gamma_0(H)$ containing all the reduced cycles in the vertex H. We denote by $V(H)$ (resp. $E(H)$) the vertex set (resp. the edge set) of $\Gamma(H)$.

By construction, $\Gamma(H)$ is admissible. Moreover, it is well known (cf. [32]) that $\Gamma(H)$ is finite if and only if H is finitely generated, $\widetilde{rk}(H) = |E(H)| - |V(H)|$ and $\Gamma(H)$ uniquely characterizes the subgroup H.

If H is finitely generated, then $\Gamma(H)$ is the positive state graph of the inverse automaton defined in section 3.1 with initial-final state H (cf. [20], [22]).

If Γ is an admissible graph then we denote by $c\Gamma$ the core of Γ. If $c\Gamma$ is empty, Γ is said to be *contractible*, equivalently Γ is a tree. In their paper Meakin an Weil represent the intersection of subgroups of $F(A)$ in terms of pull-backs in the category of $A-$labeled graphs (cf. [32]). More precisely, if H and K are finitely generated subgroups they consider the graph $\Delta(H,K)$ given by the set of vertices $V(H)\times V(K)$ and, whose a-labeled edges are of the form $((Hx,Ky),a)=((Hx,a),(Ky,a))$. As it is easy to see, the pull back $\Delta(H,K)$ is the product of the monoidal automata associated to the graphs $\Gamma(H)$ and $\Gamma(K)$ with initial-final state H and K, respectively. We denote by V and E the set of vertices of such graph.

It was observed in (cf. [24], [33]) that

$$\sum_{HgK\in H\backslash /K}\widetilde{rk}(H^g\cap K)=\sum(|E|-|V|)$$

It follows that proving the SHN is equivalent to proving that

$$\sum(|E|-|V|)\leq\widetilde{rk}(H)\widetilde{rk}(K)$$

The main part of the proof of Meakin and Weil is made under the hypothesis of a binary alphabet, $A=\{a,b\}$. Then the result is shown to be true also for a general alphabet. In the case of an admissible graph on a binary alphabet every vertex has at most two edges going out from it. So one can distinguish the vertices of such graph in *positive* if there are two edges going out from them, *negative* if there is only one edge going out from them and *zero* if there is no edge going out from them.

Denoting by P and N the sets of positive and negative vertices of $\mathcal{A}$, we have that

$$|E|-|V|=\sum_{x\in Q}n_x=|P|-|N|$$

where, for each $x\in Q$, n_x is the number of edges going out from x.

Then they consider two finitely generated subgroups H and K, their graphs $\Gamma(H)$ and $\Gamma(K)$ and the union of the non-contractible connected components of $\Delta(H,K)$. Making some consideration on the number of positive and negative vertices in the pull-back of $\Gamma(H)$ and $\Gamma(K)$ with respect to such numbers in $\Gamma(H)$ and $\Gamma(K)$ they obtain:

Theorem 3.2.1 *Let H and K be finitely generated subgroups of $F(A)$ and let n denote the number of negative vertices in Γ_H. Then*

$$\sum_{HgK\in H\backslash /K}\widetilde{rk}(H^g\cap K)\leq(\widetilde{rk}(H)+n)\widetilde{rk}(K)$$

If H is positively generated then Γ_H has no negative vertices so SHN holds.

As we have noted $\Gamma(H)$ is the positive state graph of an inverse automaton on the symmetrized alphabet $(A \cup A^{-1})$ associated to H, seen in section 3.1, and it is, moreover, a monoidal automaton. The pull back $\Delta(H, K)$ is the product of such monoidal automata $\Gamma(H)$ and $\Gamma(K)$. The connected component of such a product in (H, K) is, in particular, the positive state graph of an inverse automaton whom the subgroup $H \cap K$ is associated to (cf. [32], [37]).

Given $X \subseteq A^*$ a finite prefix set, we can consider the automaton obtained by applying the subset construction to the flower automaton of X (cf. [2]) and the inverse automaton associated to $< X >$.

Let us denote by $\mathcal{A}_{X^*}$ the automaton obtained applying the subset construction to the flower automaton of X and by $\mathcal{A}_{<X>}$ the positive state graph of the inverse automaton associated to $< X >$. They are both automata in A.

Given X, $\mathcal{A}_{<X>}$ is obtained from $\mathcal{A}_{X^*}$ by identifying the end vertices of two edges with the same label and the starting vertices of two edges with the same label and then deleting all the vertices of degree one.

In general $\mathcal{A}_{<X>}$ is different from $\mathcal{A}_{X^*}$ so the technique used in proving the result of Hanna Neumann for positively generated subgroups cannot be used for submonoids. Moreover, if $\mathcal{A}_{<X>}$ is equal to $\mathcal{A}_{X^*}$ then, as seen in example 3.7, it can happen that $rk(< X >) \neq rk(X^*)$.

The figures relative to the following two examples can be found in section 3.5.

Example 3.2.2 *In figure 3.19 there are $\mathcal{A}_{X^*}$ and $\mathcal{A}_{Y^*}$, with*

$$X = \{aab, aba\} \quad \text{and} \quad Y = \{a, baaba\}$$

and, in figure 3.20, there is their product automaton. One has that

$$X^* \cap Y^* = \{a(abaaba)^* baaba\}^*$$

In figure 3.21 there are the positive state graph of the inverse automata relative to $< X >$ and $< Y >$ and, in figure 3.22, their product automaton. One has that

$$< X > \cap < Y >=< aabaab, abaaba >$$

In this case $\mathcal{A}_{Y^*}$ is different from $\mathcal{A}_{<Y>}$. Moreover $X^* \cap Y^*$ is infinitely generated while $< X > \cap < Y >$ is finitely generated.

Example 3.2.3 *In figure 3.23 there are $\mathcal{A}_{X^*}$ and $\mathcal{A}_{Y^*}$, with*

$$X = A^2 \quad \text{and} \quad Y = A^3$$

and, in figure 3.24 there is their product automaton. One has that

$$X^* \cap Y^* = A^{6^*}$$

Moreover

$$\widetilde{rk}(X^* \cap Y^*) = 2^6 - 1 = 63 > \widetilde{rk}(X^*)\widetilde{rk}(Y^*) = (2^2 - 1)(2^3 - 1) = 21$$

In figure 3.25 there are the positive state graph of the inverse automata relative to $< X >$ and $< Y >$ and, in figure 3.26 there is their product automaton. One has that

$$< X >=< aa, ba^{-1}, bb > \quad \text{and} \quad < Y >=< a^3, a^2b, ab^{-1}, aba^{-2} >$$

and

$$< X > \cap < Y >=< ab^{-1}, aba^{-2}, a^2ba^3, aaaba^{-4}, a^4ba^{-5}, a^5b, a^6 >$$

In this case $\mathcal{A}_{Y^*}$ is different from $\mathcal{A}_{<Y>}$, and

$$\widetilde{rk}(< X > \cap < Y >) = 6 = \widetilde{rk}(< X >)\widetilde{rk}(< Y >)$$

3.3 Nielsen bases and strongly Nielsen bases: characterization and properties

In this section we recall definitions and properties of the Nielsen reduced bases and the strongly Nielsen reduced bases of a subgroup of a free group. These are bases in which the cancellation in the product of elements is reduced.

Let A be an alphabet and let $F(A)$ be the free group on A. In section 1.4 we have defined the Nielsen reduced sets and the strongly Nielsen reduced sets. We recall here the definitions:

Definition 3.3.1 *$U \subseteq F(A)$ is a Nielsen reduced set if:*

1)for each $u \in U$, $u^{-1} \notin U$

2)for each $u, v, w \in U^{\pm}$, $uv \neq 1$ and $vw \neq 1$, it is $|uvw| > |u| - |v| + |w|$

A Nielsen reduced set is a set such that in every product uvw not all of v is cancelled. This concept will be exploited in the following theorem that characterizes Nielsen reduced sets (cf. [14]). Before, let us define the imt-factorization of a set of $F(A)$.

Let U be a finite set of $F(A)$. We say that the elements of $U^{\pm}$ have imt-factorization if, each $u \in U^{\pm}$, seen as a reduced word in $(A \cup A^{-1})^*$, can be written as a product $u = i(u)m(u)t(u)$ where each $m(u)$ is non-empty and $i(u^{-1}) = t(u)^{-1}$, $m(u^{-1}) = m(u)^{-1}$ and $t(u^{-1}) = i(u)^{-1}$. Now we can give the characterization of the Nielsen reduced sets (cf. [14]):

Theorem 3.3.2 *Let $U \subseteq F(A)$ such that, for each $u \in U$, $u^{-1} \notin U$. The following are equivalent:*

1) U is Nielsen reduced

2) The elements of $U^{\pm}$ have imt-factorizations and for each $u_1, u_2 \in U^{\pm}$ such that $u_1 \neq u_2^{-1}$ reducing the product $u_1u_2 = i(u_1)m(u_1)t(u_1)i(u_2)m(u_2)t(u_2)$ consists only in reducing the factors $t(u_1)i(u_2)$.

3) The elements of $U^{\pm}$ have imt-factorizations and for each $u_1, \ldots, u_n \in U^{\pm}$ such that, for each j, $u_j \neq u_{j+1}^{-1}$ we have

$$|u_1 \cdots u_n| = |i(u_1)m(u_1)| + |t(u_1)i(u_2)| + |m(u_2)| + \ldots + |t(u_{n-1})i(u_n)| + |m(u_n)t(u_n)|$$

4) The elements of $U^{\pm}$ have imt-factorizations and for each $u_1, \ldots, u_n \in U^{\pm}$ such that, for each j, $u_j \neq u_{j+1}^{-1}$ we have

$$|u_1 \cdots u_n| = |u_1 \ldots u_{j-1}i(u_j)| + |m(u_j)| + |t(u_j)u_{j+1} \ldots u_n|$$

The previous characterization asserts that U is a Nielsen reduced set if, and only if, every element u of $U^{\pm}$, seen as reduced word, can be written as the product of three factors in such a way that the middle factor is never completely cancelled in a product with elements of $U^{\pm}$.

Example 3.3.3 *The set $U = \{aaab, aaaa, aaba^{-1}\}$ is Nielsen reduced. In fact, the elements of $U^{\pm}$ have the following imt-factorization:*

$$i(aaaa) = aaa, \;\; m(aaaa) = a \;\; t(aaaa) = 1$$

$$i(aaab) = aaa, \;\; m(aaab) = b \;\; t(aaab) = 1$$

$$i(aaba^{-1}) = aab, \;\; m(aaba^{-1}) = a^{-1} \;\; t(aaba^{-1}) = 1$$

$$i(a^{-4}) = 1, \;\; m(a^{-4}) = a^{-1} \;\; t(a^{-4}) = a^{-3}$$

$$i(b^{-1}a^{-3}) = 1, \;\; m(b^{-1}a^{-3}) = b^{-1} \;\; t(b^{-1}a^{-3}) = a^{-3}$$

$$i(ab^{-1}a^{-2}) = 1, \;\; m(ab^{-1}a^{-2}) = a \;\; t(ab^{-1}a^{-1}a^{-1}) = b^{-1}a^{-2}$$

This imt-factorization verifies property 2) of theorem 3.3.2.

Let us define now the strongly Nielsen reduced sets. These are, in particular, Nielsen reduced sets with a condition on the cancellation in the product of two elements of $F(A)$.

Definition 3.3.4 *$U \subseteq F(A)$ is a strongly Nielsen reduced set if:*

1)for each $u \in U$, $u^{-1} \notin U$

2)for each $u, v, w \in U^{\pm}$, $uv \neq 1$ and $vw \neq 1$, it is $|uvw| > |u| - |v| + |w|$

3)for each $u, v \in U^{\pm}$, $uv \neq 1$, it is $|uv| \geq max(|u|, |v|)$

For simplicity of notation, we will often call a strongly Nielsen reduced set (resp. Nielsen reduced set) simply a strongly Nielsen set (resp. Nielsen set). Let us recall a proposition, already seen in section 1.3, concerning cancellations in a product of two elements in $F(A)$:

Proposition 3.3.5 *Let $u, v \in F(A)$. There exist unique $u_1, v_1, u_2 \in F(A)$ such that $u = u_1 u_2$, $v = {u_2}^{-1} v_1$ and the word $uv = u_1 v_1$ is reduced.*

A strongly Nielsen reduced set has the property that, in every product of two elements u, v in $F(A)$, uv, it is not canceled more than the half of v and than the half of u. This is equivalent to the fact that, given u_1, v_1, u_2 as in prop. 3.3.5, $|u_2| \leq |v_2|$ or $|u_2| \leq |u_1|$, as it is verified in the following corollary:

Corollary 3.3.6 *Let $U \subseteq F(A)$ ($1 \notin U$). Let $u, v \in U^{\pm}$ and $u_1, v_1, u_2 \in F(A)$ such that $u = u_1 u_2$, $v = {u_2}^{-1} v_1$ and the word $uv = u_1 v_1$ is reduced. Then*
1)$|uv| \geq |u| \iff |u_2| \leq |v_1|$
2)$|uv| \geq |v| \iff |u_2| \leq |u_1|$.

Proof. Let $u, v \in U^{\pm}$ and $u_1, v_1, u_2 \in F(A)$ such that $u = u_1 u_2$, $v = {u_2}^{-1} v_1$ and the word $uv = u_1 v_1$ is reduced. let us prove the statement 1).

$\Longrightarrow$ Let us suppose $|uv| \geq |u|$. We have that $|uv| = |u_1| + |v_1| \geq |u| = |u_1| + |u_2|$ then $|u_2| \leq |v_1|$.

$\Longleftarrow$ Let us suppose that $|u_2| \leq |v_1|$. We have that $|uv| = |u_1| + |v_1| \geq |u| = |u_1| + |u_2|$ then $|uv| \geq |u|$.

The statement 2) is proved analogously. □

Example 3.3.7 *$U = \{aaab, aaaa, aaba^{-1}\}$ is a Nielsen reduced set but it is not a strongly Nielsen reduced set since $|b^{-1}a^{-1}a^{-1}a^{-1}aaaa| = 2 < |aaaa|$.*

3.4 Algorithms for finding strongly Nielsen bases using automata

Given an inverse automaton $\mathcal{A}$, to every spanning tree of its positive state graph it corresponds a basis of the subgroup associated to $\mathcal{A}$. It is known that such bases are Nielsen reduced bases (cf. [14]) and we prove this fact in section 3.4.1. But not all of them are strongly Nielsen bases. So, given an inverse automaton $\mathcal{A}$ and its positive state graph $\Gamma_{\mathcal{A}}$, the following questions arise:

1)Does exist a spanning tree of $\Gamma_{\mathcal{A}}$ whose associated basis is a strongly Nielsen basis?

2)Given a spanning tree of $\Gamma_{\mathcal{A}}$, how can we construct starting from it another tree whose associated basis is a strongly Nielsen basis?

We answer these questions in the following subsection 3.4.1. Instead, in subsection 3.4.2, we prove that the breadth-first tree (see section 1.4.2) is a tree whose basis is strongly Nielsen. As an application of such result, in subsection 3.4.3, we construct an algorithm which, given a finite set of elements in $F(A)$, allows us to construct an equivalent set that is strongly Nielsen. The complexity of such algorithm turns out to be of interest.

3.4.1 A first algorithm for the construction of a strongly Nielsen basis

In this subsection we start by proving the result (cf. [14]) stating that, given an inverse automaton, to every spanning tree of its positive state graph it is associated a Nielsen reduced basis.

Such a basis is not necessarily a strongly Nielsen basis, so in the second part of the section, starting from a spanning tree we construct another spanning tree whose associated basis is a strongly Nielsen reduced basis.

Let $\mathcal{A} = (Q, i, i, \mathcal{E} \cup \mathcal{E}^{-1})$ be an inverse automaton on $A \cup A^{-1}$, $\Gamma_{\mathcal{A}} = (Q, \mathcal{E})$ its positive state graph and $H = red(L(\mathcal{A}))$ the subgroup associated to $\mathcal{A}$.
We have seen, in section 3.1, that for each spanning tree of $\Gamma_{\mathcal{A}}$, $T = (Q, \mathcal{F}(T))$, we get a basis of $red(L(\mathcal{A}))$, that is $\ell(B_T)$ the set of labels of the set

$$B_T = \{b_e \ , \ \ \forall e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})\}$$

where $b_e = p_{i(e)} e p_{f(e)}^{-1}$ and $\widetilde{T} = (Q, \mathcal{F}(\widetilde{T}))$ is the directed graph associated to T.

We have seen, in section 3.1.2, that such a path is reduced and so, by prop. 3.1.9, $\ell(b_e)$ is a reduced word.

Remark 3.4.1 *Let us note that $\ell(B_T)^{\pm} = \ell(B_T{}^{\pm})$. For each $b_e \in B_T$, one has that b_e is a path in $\mathcal{A}$ from i to i with label $\ell(b_e)$. Since $\mathcal{A}$ is inverse ${b_e}^{-1}$ is a path in $\mathcal{A}$ with label $\ell(b_e)^{-1}$ so $\ell({b_e}^{-1}) = \ell(b_e)^{-1}$.*

Let us define now two functions characterizing each element in ${B_T}^{\pm}$:
Let

$$\psi : {B_T}^{\pm} \longrightarrow \mathcal{E} \setminus \mathcal{F}(\widetilde{T})$$

such that, for each $b_l \in B_T$, $\psi(b_l) = l$ and, for each ${b_l}^{-1} \in {B_T}^{-1}$, $\psi(b_l^{-1}) = l$.
Let

$$\iota : {B_T}^{\pm} \longrightarrow \{1, -1\}$$

such that, for each $b_l \in B_T$, $\iota(b_l) = 1$ and, for each ${b_l}^{-1} \in {B_T}^{-1}$, $\iota(b_l^{-1}) = -1$.

It is well known that $\ell(B_T)$ is a Nielsen reduced basis. We present here a proof of this result. In particular, in the next proposition we prove that the elements of $\ell(B_T)^{\pm}$ have an imt-factorization satisfying property 2) of theorem 3.3.2.

Proposition 3.4.2 *Let $\mathcal{A}$ be an inverse automaton, $\Gamma_{\mathcal{A}}$ its positive state graph, and T a spanning tree of $\Gamma_{\mathcal{A}}$. Then $\ell(B_T)$ is a Nielsen reduced set.*

Proof. Let $u, v \in \ell({B_T}^{\pm})$ such that $u \neq v^{-1}$. Let $c, d \in {B_T}^{\pm}$ such that $u = l(c)$ and $v = l(d)$. Let $\psi(c) = e$ and $\psi(d) = g$. Let us begin by proving the following:
-There exist $\mathbf{p}, \mathbf{r}, \mathbf{q}$ paths in $\mathcal{A}$ such that $\mathbf{c = pr}$, $\mathbf{d = r^{-1}q}$

By prop. 3.3.5, there exist unique $u_1, u_2, v_1 \in F(A)$ such that $u = u_1 u_2$, $v = {u_2}^{-1} v_1$ and $u_1 v_1$ is reduced. Let r be the suffix path of c in $\mathcal{A}$ with label u_2,

$$r : x \xrightarrow{u_2} i$$

Since $\mathcal{A}$ is an inverse automaton (see prop. 3.1.1), then ${u_2}^{-1}$ is the label of the path r^{-1} from i to x and so r^{-1} is the prefix path of d with label ${u_2}^{-1}$. Let p be the path with label u_1 starting at i and q the path with label v_2 starting at x, then

$$c = pr, \quad d = r^{-1}q$$

We will prove now that the elements of $\ell(B_T)^{\pm}$ have an imt-factorization .
-The elements of $\ell(\mathbf{B_T})^{\pm}$ have an imt-factorization.

Let us consider the following factorization in $\ell(B_T)^{\pm}$: for each $u \in \ell(B_T)^{\pm} = \ell({B_T}^{\pm})$, if $u = \ell(b_e)$ with $e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})$ then

$$i(u) = \ell(p_{i(e)}), \;\; m(u) = \ell(e), \;\; t(u) = \ell(p_{f(e)}^{-1})$$

If $u = \ell({b_e}^{-1})$ with $e \in \mathcal{F} \setminus \mathcal{F}(T)$ then

$$i(u) = \ell(p_{f(e)}), \;\; m(u) = \ell(e^{-1}), \;\; t(u) = \ell({p_{i(e)}}^{-1})$$

It can be easily verified that such factorization is an imt-factorization.
One has that, for each $u \in \ell({B_T}^{\pm})$, the paths correspondent to $i(u)$ and $t(u)$ are contained in T.

Next we prove property 2) of theorem 3.3.2 that characterizes the Nielsen reduced sets.

- Reducing the product uv = i(u)m(u)t(u)i(v)m(v)t(v) consists only in reducing the factor t(u)i(v).

In $c = pr$ and $d = r^{-1}q$ the unique edge not contained in T is e and g, respectively. If reducing uv does not consist only in reducing $t(u)i(v)$ then either $u_2 = u_3 t(u)$, with u_3 not empty word, or ${u_2}^{-1} = i(v)v_3$, with v_3 not empty word.

Let us suppose that $u_2 = u_3 t(u)$, with u_3 not empty word. Then the path correspondent to $m(u)t(u)$ is a suffix of r. So we have that e occurs in r. If $u \neq v$ then this is a contradiction since $e \neq g$, being $u \neq v^{-1}$, and the unique edge in d not contained in T is g.

We will show now that if $u = v$ then $c = r^{-1}p'r$, with p' not null path. In fact $c = pr = r^{-1}q$ and one has that r is a proper suffix of q otherwise either $q = r$ and c is not reduced or $r = r_1 q$, with r_1 not null. In the last case since $pr = r^{-1}q$ one has that $r^{-1} = r_0 r_1$ and since $r = r_1 q$ then $r^{-1} = r_0 r_1 = q^{-1}{r_1}^{-1}$ that is a contradiction since the unique path equal to its inverse is the null path and r_1 is not the null path.

So $c = r^{-1}p'r$, with p' not null path, and c contains twice e and this is a contradiction since e is contained in c only once.

It is proved analogously if we suppose that ${u_2}^{-1} = i(v)v_3$, with v_3 not empty word. □

As we have seen, to every spanning tree T of $\Gamma_{\mathcal{A}}$ it corresponds a Nielsen reduced basis. This basis is not necessarily a strongly Nielsen basis as we can see in the following example:

Example 3.4.3 *In figure 3.11 we have the positive state graph of an inverse automaton $\mathcal{A}$. The edges underlined are those ones of T spanning tree. One has that $l(B_T) = \{aaab, aaaa, aaba^{-1}\}$ is not a strongly Nielsen reduced basis since $|a^{-1}a^{-1}a^{-1}a^{-1}aaab| = |a^{-1}b| = 2 < |aaaa|$.*

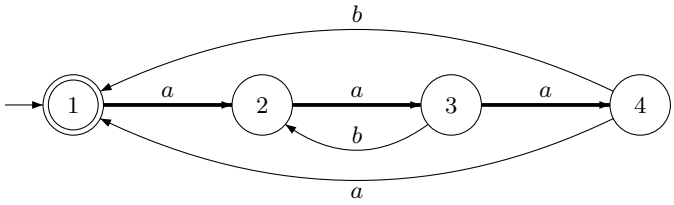

Figure 3.11:

It is natural to ask ourself which are the trees corresponding to strongly Nielsen reduced sets and how to catch such trees. Here we present an algorithm that, given an arbitrary spanning tree of $\Gamma_{\mathcal{A}}$, builds a spanning tree whose corresponding basis is a

strongly Nielsen basis. For this purpose, let us define the score of a spanning tree. It will give us the worst number of steps in building such a tree from a fixed one.

Definition 3.4.4 *Let $T = (Q, \mathcal{F}(T))$ be a spanning tree of $\Gamma_{\mathcal{A}} = (Q, \mathcal{E})$ and let $\widetilde{T} = (Q, F(\widetilde{T}))$ the directed graph associated to it. We define $S(T) = \sum_{e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})} |b_e|$ as the score of T.*

One has that $|b_e| = |\ell(b_e)|$ and so $S(T) = \sum_{e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})} |\ell(b_e)|$ is the sum of the lengths of the elements in the basis $\ell(B_T)$.

Example 3.4.5 *In the example in figure 3.11, $S(T) = |aaab| + |aaaa| + |aaba^{-1}| = 12$.*

Let now $\mathcal{A}$ be an inverse automaton and let $T = (Q, \mathcal{F}(T))$ be a spanning tree of $\Gamma_{\mathcal{A}}$ such that $\ell(B_T)$ is not a strongly Nielsen reduced basis. So there exist $u, v \in \ell(B_T)^{\pm}$ with $u \neq v^{-1}$ such that either $|uv| < |u|$ or $|uv| < |v|$. Let us note that one has $u \neq v$ since $|uu| \geq |u|$. Let us suppose that $|uv| < |u|$, we will see that the other case is reduced to this one.
By prop. 3.3.5 there exist unique $u_1, u_2, v_1 \in F(A)$ such that $u = u_1 u_2$, $v = (u_2)^{-1} v_1$ and $u_1 v_1$ is reduced. Moreover, by prop. 3.3.6, $|u_2| > |v_1|$. We find a similar factorization for the paths of $\mathcal{A}$ whose labels are u and v, as it is shown in the next proposition.

Proposition 3.4.6 *Let $u, v \in \ell(B_T)^{\pm}$, $u = \ell(c)$ and $v = \ell(d)$ with $c, d \in {B_T}^{\pm}$, such that $u \neq v^{-1}$ and $|uv| < |u|$. Then there exist p, q, r paths in $\mathcal{A}$ such that*

1. *$c = pr$ and $d = r^{-1}q$*
2. *r is contained in T*
3. *$|r| > |q|$ and pq is reduced*

Proof.
(1)**There exist $\mathbf{p, r, q}$ paths in $\mathcal{A}$ such that $\mathbf{c = pr}$, $\mathbf{d = r^{-1}q}$**
It follows by the proof of prop. 3.4.2.
(2)**r is contained in T**
Let $\psi(c) = e$ and $\psi(d) = g$. Since $u \neq v$ and $u \neq v^{-1}$ then $e \neq g$. As already observed in the proof of prop. 3.4.2, in c and d the unique edge not contained in T is e and g, respectively. If, by contradiction, r is not contained in T then e or g is contained in r. Let us suppose e contained in r, then d contains two different edges not contained in T, contradiction! It is proved analogously if we suppose g contained in r.
(3)**$|\mathbf{r}| > |\mathbf{q}|$ and pq is reduced.**
From $|uv| < |u|$ it follows that $|u_2| > |v_1|$. By prop. 3.1.9, one has that $|r| = |u_2| > |v_1| = |q|$. The path pq has, by construction, a reduced label. So, by prop. 3.1.9 pq is reduced. □

Example 3.4.7 *In the example in figure 3.11, taking as* $u = a^{-1}a^{-1}a^{-1}a^{-1}$ *and* $v =$ *aaab one has in* $\mathcal{A}$

$$p : 1 \xrightarrow{a^{-1}} 4$$

$$r : 4 \xrightarrow{a^{-1}} 3 \xrightarrow{a^{-1}} 2 \xrightarrow{a^{-1}} 1$$

$$q : 4 \xrightarrow{b} 1$$

From now on, unless otherwise stated, we assume to be under the following hypotheses:
Let $u, v \in \ell(B_T)^{\pm}$, $u = \ell(c)$ and $v = \ell(d)$ with $c, d \in {B_T}^{\pm}$, such that $u \neq v^{-1}$ and $|uv| < |u|$. Let $\psi(c) = e$ and $\psi(d) = g$.
As in prop. 3.4.6, let p, q, r paths in $\mathcal{A}$ such that

$$c = pr \quad d = r^{-1}q$$

r is contained in T, $|r| > |q|$ and pq is reduced.
Let p_1, p_2, q_1, q_2, r paths in $\mathcal{A}$ such that

$$\text{either} \quad c = p_1 e p_2 r \ \text{ or } \ c = p_1 e^{-1} p_2 r$$

and

$$\text{either} \quad d = r^{-1} q_1 g q_2 \ \text{ or } \ r^{-1} q_1 g^{-1} q_2$$

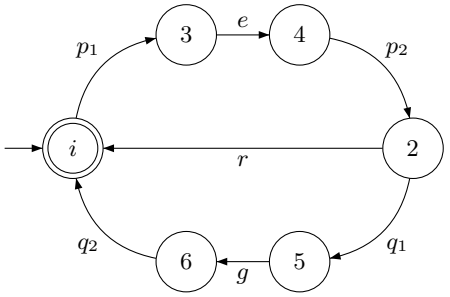

Figure 3.12: Structure of a subgraph of $\Gamma_{\mathcal{A}}$ correspondent to $c = p_1 e p_2 r$ and $d = r^{-1} q_1 g q_2$

Now we want to construct, starting from T, a spanning tree T' which has a score lower than that one of T. In particular letting $r = hr_1$, with $h \in \mathcal{E}$, we will consider the graph obtained from T by cutting the edge $\overline{h}$ and adding the edge $\overline{g}$, that is correspondent to v.

For this purpose we will prove that $\mathcal{A}$ has a particular shape, that is r cannot be a suffix of d and h is not contained in q.

Then we will prove that such graph T' is still a spanning tree of $\Gamma_{\mathcal{A}}$ with score lower than that one of T.

Let us observe now that r cannot be a suffix of d.

Proposition 3.4.8 *r is not a suffix of d.*

Proof. By prop. 3.4.6, $|r| > |q|$. So, r cannot be a suffix of d. □

This means in particular that in $\Gamma_{\mathcal{A}}$ it is not possible to have the subgraph as in figure 3.13.

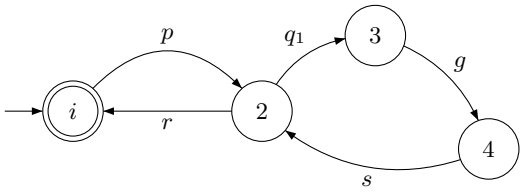

Figure 3.13: Structure of a subgraph of $\Gamma_{\mathcal{A}}$ correspondent to $c = pr$ and $d = r^{-1}q_1gsr$

Proposition 3.4.9 *Every edge occurring in p_2r occurs just once. Every edge occurring in $r^{-1}q_1$ occurs just once.*

Proof. It follows from the fact that they are reduced paths contained in T. □

Let now h be the first edge of r in $\mathcal{A}$, $r = hr_1$ with $h \in \mathcal{E}$. In the following proposition we prove that h does not occur in q:

Proposition 3.4.10 *The edge h does not occur in q.*

Proof. By prop. 3.4.8 r is not a suffix of d. Then r is not a suffix of q, where either $q = q_1gq_2$ or $q = q_1g^{-1}q_2$. Let us prove that h does not occur in q_1 and does not occur in q_2.

-h does not occur in q_1
Since $r^{-1}q_1$ is a reduced path contained in T and since h occurs in r then, by prop. 3.4.9, h does not occur in q_1.

-h does not occur in q_2
Let us consider the reduced path associated to $rq_2{}^{-1}$ in $\mathcal{A}$. Since r is not a suffix of d, and so of q_2, then $red(rq_2{}^{-1})$ contains h. Since $red(rq_2{}^{-1})$ is a reduced path contained in T then, by prop. 3.1.18, it contains h only once and so h does not occur in q_2. □

Proposition 3.4.11 *If r^{-1} is not a prefix of c, then h does not occur in p.*

Proof. The proof is analogous to that one of the prop. 3.4.10. □

Let us consider $T' = (Q(T'), \mathcal{F}(T'))$ the tree obtained from T by adding $\overline{g}$ and cutting $\overline{h}$ from the set of edges, that is $\mathcal{F}(T') = (\mathcal{F}(T) \setminus \{\overline{h}\}) \cup \{\overline{g}\}$. The set of vertices $Q(T')$ is the set of vertices induced by the definition of $\mathcal{F}(T')$.

Example 3.4.12 *In the example in figure 3.14 we have $\Gamma_{\mathcal{A}}$ the positive state graph of an inverse automaton. The edges underlined are those ones of a spanning tree T' obtained from T, spanning tree in figure 3.11, by adding the edge $g : 4 \xrightarrow{b} 1$ and cutting the edge $h : 3 \xrightarrow{a} 4$. One has that $\ell(B_{T'}) = \{aaab, b^{-1}a, aaba^{-1}\}$ and $S(T') = 10 < S(T)$.*

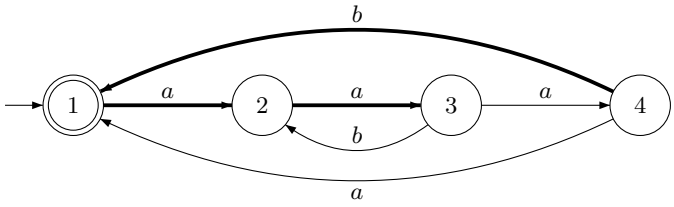

Figure 3.14: $\Gamma_{\mathcal{A}}$ positive state graph of an inverse automaton.

Proposition 3.4.13 *Let $T' = (Q(T'), \mathcal{F}(T'))$ be such that $\mathcal{F}(T') = (\mathcal{F}(T) \setminus \{\overline{h}\}) \cup \{\overline{g}\}$ and $Q(T')$ is the set of vertices induced by the definition of $\mathcal{F}(T')$. It follows that $Q(T') = Q(T) = Q$.*

Proof. Let us prove that $i(h), f(h) \in Q(T')$. So we will have that $Q(T') \supseteq Q(T)$ and since $Q(T) = Q$ and $Q(T') \subseteq Q$ then $Q(T') = Q(T) = Q$.

So we have to prove that there exist two edges in $\mathcal{F}(T')$ having respectively $i(h)$ and $f(h)$ as one of the extremal vertices. We will prove that there exist two paths in $\widetilde{T'}^{inv}$ with this property.

The path q^{-1} starts at i and ends at $i(h)$. It is contained in T' since q_1 and q_2 are contained in T, h does not occur in q and g is contained in T', by definition. Since q^{-1} is a not null path and the final state of q^{-1} is $i(h)$, then $i(h) \in Q(T')$. One has that $r = hr_1$. Moreover we have that r_1 is not a null path otherwise $r = h$ and, by prop. 3.4.6, $1 = |h| > |q| \geq 1$, that is a contradiction. So we have that $i(r_1) = f(h)$, with r_1 path contained in T' since it is contained in T and h does not occur in r_1. Finally we get that $Q(T') = Q(T) = Q$. □

In the next proposition we prove that T' is a spanning tree of $\Gamma_{\mathcal{A}}$ and that the score of T' is less than the score of T.

Proposition 3.4.14 *Let $T' = (Q(T'), \mathcal{F}(T'))$ be such that $\mathcal{F}(T') = (\mathcal{F}(T) \setminus \{\overline{h}\}) \cup \{\overline{g}\}$ and $Q(T')$ is the set of vertices induced by the definition of $\mathcal{F}(T')$. Then T' is a spanning tree of $\Gamma_{\mathcal{A}}$.*

Proof.

1) $|\mathcal{F}(T')| = |\mathcal{F}(T)|$ and $|Q(T')| = |Q(T)|$.

The first statement is trivial. The second one follows from prop. 3.4.13.

2)T' is connected.

Let $x, y \in Q(T')$, we want to show that there is a path in T' from x to y. We will prove it by showing that there is a path from x to y in $(\widetilde{T'})^{inv}$. Since T is connected then let s be the reduced path contained in T from x to y. If h does not occur in s then s is contained in T'.

If h occurs in s then either $s = s_1 h s_2$ or $s = s_1 h^{-1} s_2$ and h does not occur in s_1, s_2 (see prop. 3.1.18), so s_1, s_2 are contained in T'. By construction $q{r_1}^{-1}$ is a path whose initial and final vertices are those ones of h and it is contained in T' since h does not occur in q, by prop. 3.4.8 and h does not occur in r_1 since $r = hr_1$ is a reduced path contained in T. So if $s = s_1 h s_2$ then we can consider $s' = s_1 q {r_1}^{-1} s_2$ that is a path from x to y contained in T'. If $s = s_1 h^{-1} s_2$ then we can consider $s' = s_1 r_1 q^{-1} s_2$ that is a path from x to y contained in T'.

3)T' is a spanning tree.

Since $|\mathcal{F}(T')| = |Q(T')| - 1$ and, by 2), T' is connected then T' is a tree. Since $Q(T') = Q$ then it is a spanning tree. □

In the following theorem we will prove that $S(T')$ is less than $S(T)$. Let us denote, for each $e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})$, with b_e^T the path $b_e \in B_T$ and for each $e \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T'})$, by $b_e^{T'}$ the path $b_e \in B_{T'}$.

Theorem 3.4.15 *Let $T' = (Q(T'), \mathcal{F}(T'))$ be such that $\mathcal{F}(T') = (\mathcal{F}(T) \setminus \{\overline{h}\}) \cup \{\overline{g}\}$. Then $S(T') < S(T)$*

Proof. We want to prove that for each $l \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T'})$, $l \neq e$, there exists an edge $l' \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})$, $l' \neq e$, such that $|b_l^{T'}| \leq |b_{l'}^T|$ and that $|b_e^{T'}| < |b_e^T|$.

One has that

$$\mathcal{E} \setminus \mathcal{F}(\widetilde{T'}) = \mathcal{E} \setminus ((\mathcal{F}(\widetilde{T}) \cup \{\overline{g}\}) \setminus \{\overline{h}\}) = (\mathcal{E} \setminus (\mathcal{F}(\widetilde{T}) \cup \{\overline{g}\})) \cup \{\overline{h}\}$$

Let us consider $h \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T'})$

-q is a reduced path contained in T'

The edge h does not occur in q. Moreover q_1, q_2 are contained in T and g is contained in T', so q is a path contained in T'. The path q is reduced since it is a subpath of ${b_g}^T$ that is a reduced path.

-r_1 is a reduced path contained in T'

Since $r = hr_1$ and r is a reduced path contained in T it follows that h does not occur in r_1 and so r_1 is contained in T'. Moreover, it is a reduced path since it is a subpath of b_e reduced path.

-So we have that

$$b_h^{T'} = q^{-1}hr_1 = q^{-1}r = b_g^T$$

hence

$$|b_h^{T'}| = |b_g^T|$$

Let us consider $e \in \mathcal{F} \setminus \mathcal{F}(T')$ and two cases:
a)r^{-1} is not a prefix of c
b)r^{-1} is a prefix of c

a) Let us suppose that r^{-1} is not a prefix of c.

-p_1 is a reduced path contained in T'

By construction p_1 is a reduced path contained in T and, by prop. 3.4.11, h does not occur in p_1. So p_1 is contained in T'.

-p_2q is a reduced path contained in T'

By prop. 3.4.10 and 3.4.11, p_2q is a path contained in T', since h does not occur in p_2 that is contained in T and q is contained in T', as already observed. Moreover p_2q is reduced since subpath of pq that is a reduced path, by prop. 3.4.6.

-If $\iota(c) = 1$ we have that

$$b_e^{T'} = p_1ep_2q = pq = red(cd)$$

hence

$$|b_e^{T'}| = |pq| = |cd| < |c| = |b_e^T|$$

-If $\iota(c) = -1$ then

$$b_e^{T'} = q^{-1}p_2^{-1}ep_1^{-1}$$

and

$$b_e^{T'} = (pq)^{-1} = red((cd)^{-1})$$

hence

$$|b_e^{T'}| = |(pq)^{-1}| = |cd| < |c| = |b_e^T|$$

b) Let r^{-1} be a prefix of c.

Since $c = r^{-1}c_1 = pr$, either $p = p_1ep_2$ or $p = p_1e^{-1}p_2$ and r is contained in T then r^{-1} is a prefix of p_1.

-If $\iota(c) = 1$ then let $b_e^T = r^{-1}setr$, for some s, t paths of $\mathcal{A}$.

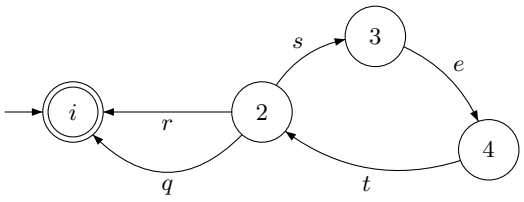

Figure 3.15: Structure of a subgraph of $\Gamma_{\mathcal{A}}$ correspondent to $c = r^{-1}setr$ and $d = r^{-1}q$

Let us consider $red(q^{-1}s)$, the reduced path associated to $q^{-1}s$. Since h occurs in r^{-1} and $r^{-1}s$ is a reduced path then h does not occur in s. Moreover h does not occur in q so $red(q^{-1}s)$ is contained in T'.

The path tq is reduced since it is a subpath of pq reduced path. Moreover it is contained in T' since h does not occur in q and in t. Finally we get:

$$b_e^{T'} = red(q^{-1}s)etq$$

and

$$|b_e^{T'}| \leq |q^{-1}s| + |et| + |q| < |r^{-1}| + |s| + |et| + |r| = |b_e^T|$$

-If $\iota(c) = -1$ then $c = (b_e^T)^{-1}$. By construction $(b_e^T)^{-1} = r^{-1}se^{-1}tr$.
We get

$$b_e^{T'} = q^{-1}t^{-1}e\ red(s^{-1}q)$$

and we have that

$$|b_e^{T'}| < |(b_e^T)^{-1}| = |b_e^T|$$

Let us consider now $l \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T'})$, $l \neq e, h$.

If $b_l^T = p_{i(l)}lp_{f(l)}^{-1}$ does not contain h then $p_{i(l)}$ and $p_{f(l)}^{-1}$ are contained in T' and

$$b_l^{T'} = b_l^T$$

Let us suppose that b_l^T contains h. Let $b_l^T = p_{i(l)}lp_{f(l)}^{-1}$. If h is in $p_{i(l)}$ then r^{-1} is a prefix of $p_{i(l)}$. If not then the path $red(rp_{i(l)})$ in T contains twice the edge h.

If h is not contained in $p_{f(l)}$ then $p_{i(l)} = r^{-1}s_1$ and $b_l^T = r^{-1}s_1lp_{f(l)}^{-1}$. We replace r^{-1} by q^{-1} and we obtain

$$b_l^{T'} = red(q^{-1}s_1)lp_{f(l)}^{-1}$$

hence

$$|b_l^{T'}| < |b_l^T|$$

If h is in $p_{f_l}^{-1}$ then, analogously as before $p_{f(l)}^{-1} = s_2 r$. Then $b_l^T = r^{-1} s_1 l s_2 r$. We replace r by q and we obtain

$$b_l^{T'} = red(q^{-1} s_1) l \; red(s_2 q)$$

hence

$$|b_l^{T'}| < |b_l^T|$$

If h occurs in $p_{f(l)}^{-1}$ and does not occur in $p_{i(l)}$ then, analogously as before, we find $b_l^T = p_{i(l)} l s r$. We replace r by q and we obtain

$$b_l^{T'} = p_{i(l)} l \; red(sq)$$

hence

$$|b_l^{T'}| < |b_l^T|$$

We have so obtained that, for each $l \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T}')$ with $l \neq h, e$, it is

$$|b_l^{T'}| \leq |b_l^T|$$

and

$$|b_h^{T'}| = |b_g^T|$$

and

$$|b_e^{T'}| < |b_e^T|$$

Finally

$$S(T') = \sum_{l \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T'})} |b_l^{T'}| = |b_h^{T'}| + |b_e^{T'}| + \sum_{l \in (\mathcal{E} \setminus \mathcal{F}(\widetilde{T})) \setminus \{g,e\}} |b_l^{T'}|$$

On the other hand

$$S(T) = \sum_{l \in \mathcal{E} \setminus \mathcal{F}(\widetilde{T})} |b_l^T| = |b_g^T| + |b_e^T| + \sum_{l \in (\mathcal{E} \setminus \mathcal{F}(\widetilde{T})) \setminus \{g,e\}} |b_l^T|$$

Since

$$|b_h^{T'}| + |b_e^{T'}| + \sum_{l \in (\mathcal{E} \setminus \mathcal{F}(\widetilde{T})) \setminus \{g,e\}} |b_l^{T'}| < |b_g^T| + |b_e^T| + \sum_{l \in (\mathcal{E} \setminus \mathcal{F}(\widetilde{T})) \setminus \{g,e\}} |b_l^T|$$

it follows that

$$S(T') < S(T)$$

that concludes the proof. □

Remark 3.4.16 *We realized the above construction under the hypothesis that* $|uv| < |u|$. *If* $|uv| < |v|$ *then we can reduce to the first case: in fact if* $|uv| < |v|$ *then one has* $|v^{-1}u^{-1}| < |v^{-1}|$, *so defining* $u' := v^{-1}$ *and* $v' := u^{-1}$ *we have that* $|u'v'| < |u'|$.

Example 3.4.17 *In figure 3.16 there is the positive state graph of an inverse automaton* $\mathcal{A}$. *The underlined edges are those ones of a spanning tree* T. *It is*

$$\ell(B_T) = \{bbab^{-1}, ba^{-1}ba^{-1}, a^3b, a^4, aba^{-3}, a^2ba^{-2}\}$$

The basis $\ell(B_T)$ *is not a strongly Nielsen basis since* $|a^{-4}a^3b| < |a^{-4}|$.

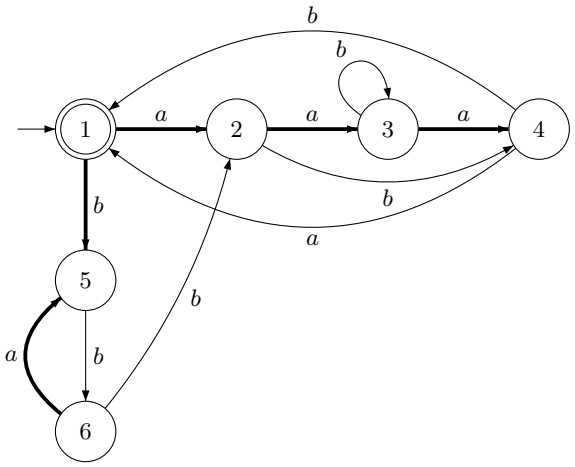

Figure 3.16: $\Gamma_{\mathcal{A}}$ positive state graph of an inverse automaton.

So we construct the tree T', *whose underlined edges are in the graph* $\Gamma_{\mathcal{A}}$ *in figure 3.17, adding the edge* $4 \xrightarrow{b} 1$ *corresponding to the element* a^3b *and cutting the edge* $3 \xrightarrow{a} 4$ *corresponding to the last* a *of* a^3. *In figure 3.16 there is the positive state graph of* $\mathcal{A}$ *with underlined the edges of* T'. *The basis associated to* T' *becomes:*

$$\ell(B_{T'}) = \{bbab^{-1}, ba^{-1}ba^{-1}, b^{-1}a, abb, a^2ba^{-2}, a^3b\}$$

As we can note, the generator a^4 *has been substituted with the generator* $b^{-1}a$ *of minor length and the generator* aba^{-3} *has been substituted with the generator* abb. *In fact we substitute all the paths in* B_T *having as suffix the path*

$$4 \xrightarrow{a^{-1}} 3 \xrightarrow{a^{-1}} 2 \xrightarrow{a^{-1}} 1$$

with the shorter path

$$4 \xrightarrow{b} 1$$

In terms of score we have:

$$S(T) = 26 > S(T') = 22$$

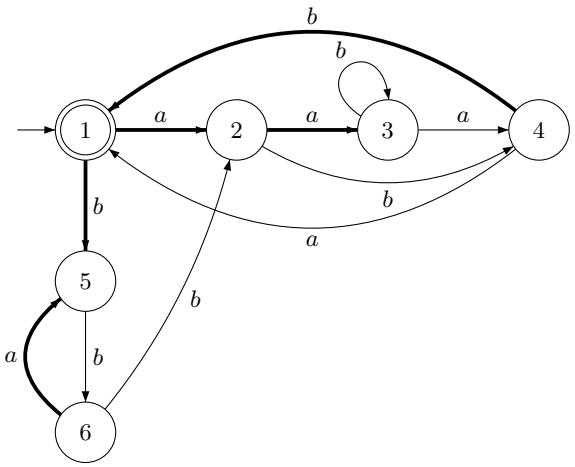

Figure 3.17: $\Gamma_{\mathcal{A}}$ positive state graph of an inverse automaton.

By these constructions we get the following theorem:

Theorem 3.4.18 *Let $\mathcal{A}$ be an inverse automaton, $\Gamma_{\mathcal{A}}$ be its positive state graph and T be a spanning tree of $\Gamma_{\mathcal{A}}$ If there exist $u, v \in \ell(B_T)^{\pm}$ such that $|uv| < max(|u|, |v|)$, then there exists a spanning tree T' such that $S(T') < S(T)$*

Now we can prove the proposition:

Theorem 3.4.19 *Let $\mathcal{A}$ be an inverse automaton and $\Gamma_{\mathcal{A}}$ its positive state graph. Let T be a spanning tree of $\Gamma_{\mathcal{A}}$ such that there exist $u, v \in \ell(B_T)^{\pm}$ such that $|uv| < max(|u|, |v|)$. Then we can construct in a finite number of steps a spanning tree $\widetilde{T}$ such that $\ell(B_{\widetilde{T}})$ is a strongly Nielsen basis.*

Proof. Let $\mathcal{A}$ be an inverse automaton. Let T be a spanning tree of $\Gamma_{\mathcal{A}}$ such that there exist $u, v \in \ell(B_T)^{\pm}$ such that $|uv| < max(|u|, |v|)$. By theorem 3.4.18, we can construct a spanning tree of $\Gamma_{\mathcal{A}}$, T' such that $S(T') < S(T)$. We iterate this procedure until we get a tree $\widetilde{T}$ such that $\ell(B_{\widetilde{T}})$ is a strongly Nielsen basis. □

Remark 3.4.20 *As we have seen, starting from a spanning tree T of the positive state graph of an inverse automaton $\mathcal{A}$ we can construct, in a finite number of steps, a spanning tree T' whose associated basis is a strongly Nielsen basis. In particular this algorithm, given a spanning tree T and so an associated Nielsen basis $\ell(B_T)$, constructs a strongly Nielsen reduced set equivalent to $\ell(B_T)$, that is $\ell(B'_T)$.*

If we consider the transformations on words for every elementary step we find that such transformations are similar to the elementary transformations of Nielsen (see section 1.2).

In fact, let $\mathcal{A} = (Q, i, i, \mathcal{F})$ *be an inverse automaton,* $\Gamma_{\mathcal{A}}$ *its positive state graph and* T *a spanning tree of* $\Gamma_{\mathcal{A}}$ *such that* $\ell(B_T)$ *is not a strongly Nielsen basis. Let so* $u, v \in \ell(B_T)^{\pm}$, $u = \ell(c)$ *and* $v = \ell(d)$ *with* $c, d \in {B_T}^{\pm}$, *such that* $u \neq v^{-1}$ *and* $|uv| < |u|$.

Let us suppose that $\psi(c) = e$ *and* $\psi(d) = g$. *Let* p, q, r *paths in* $\mathcal{A}$, *as in prop. 3.4.6, such that*

$$c = pr \quad d = r^{-1}q$$

Let $r = hr_1$ *with* $h \in \mathcal{F}$.
To $T = (Q, \mathcal{F}(T))$ *we associate* $T' = (Q, \mathcal{F}(T'))$, *where* $\mathcal{F}(T') = (\mathcal{F}(T) \setminus \{\overline{h}\}) \cup \{\overline{g}\}$. *Let us see now how the words associated to every edge not contained in* T *are transformed in words associated to every edge not contained in* T'.

$$d = b_g^T \longrightarrow b_h^{T'} = d$$

If r^{-1} *is not a prefix of* c *then, if* $\iota(c) = 1$

$$c = b_e^T \longrightarrow b_e^{T'} = cd$$

and if $\iota(c) = -1$

$$c = b_e^T \longrightarrow b_e^{T'} = d^{-1}c^{-1}$$

If r^{-1} *is a prefix of* c *then, if* $\iota(c) = 1$

$$c = b_e^{T'} \longrightarrow b_e^{T'} = d^{-1}cd$$

and if $\iota(c) = -1$

$$c = b_e^{T'} \longrightarrow b_e^{T'} = d^{-1}c^{-1}d$$

For each l *not contained in* T', $l \neq e, h$, *if* r^{-1} *is not a prefix of* b_l^T *and* r *is not a suffix of* b_l^T *then*

$$b_l^T \longrightarrow b_l^{T'}$$

if r^{-1} *is a prefix of* b_l^T *and* r *is not a suffix of* b_l^T *then*

$$b_l^T \longrightarrow b_l^{T'} = d^{-1}b_l^T$$

if r^{-1} *is not a prefix of* b_l^T *and* r *is a suffix of* b_l^T *then*

$$b_l^T \longrightarrow b_l^{T'} = b_l^T d$$

if r^{-1} *is a prefix of* b_l^T *and* r *is a suffix of* b_l^T *then*

$$b_l^T \longrightarrow b_l^{T'} = d^{-1}b_l^T d$$

Such transformations, as we can see, are strictly close to the Nielsen transformations. Moreover in an elementary step one can do more than one transformation while usually the algorithms that work on words utilize in an elementary step just one elementary Nielsen transformation.

3.4.2 A second algorithm for the construction of a strongly Nielsen basis

In the previous subsection we have seen how to construct, starting from any tree of the positive state graph of an inverse automaton, a tree whose associated basis is a strongly Nielsen basis. Here we want to show that, given the positive state graph of an inverse automaton, how one can suddenly find a tree whose basis is a strongly Nielsen basis. Such a tree is the breadth-first tree.

In particular, this tree is obtained applying the breadth-first search (BFS in short) to the positive state graph in the initial final state (see section 1.4).

Let us briefly recall the properties of such a tree:

Proposition 3.4.21 *Let $\Gamma = (V, E)$ be an undirected multigraph. Let $s \in V$. The breadth-first tree $T = (V', E')$ of Γ in s, obtained applying the BFS to Γ in s, is such that:*

1) T is a tree

2) V' consists of all vertices reachable from s

3) For all $v \in V'$ there is a unique simple path from s to v in T, that is also one of the shortest paths from s to v in Γ.

Let now $\mathcal{A}$ be an inverse automaton and let $\Gamma_{\mathcal{A}} = (Q, \mathcal{E})$ be its positive state graph. Let $\Gamma'_{\mathcal{A}}$ be the undirected multigraph associated to $\Gamma_{\mathcal{A}}$. Let T_{BFS} be the breadth-first tree of $\Gamma'_{\mathcal{A}}$ in i. Since $\mathcal{A}$ is trim then T_{BFS} is a spanning tree of $\Gamma'_{\mathcal{A}}$. So, for all $x \in Q$ there is a unique simple path from i to x in T_{BFS}, that is also one of the shortest paths from i to x in $\Gamma'_{\mathcal{A}}$.

Let us see prove now that the basis of $red(L(\mathcal{A}))$ associated to T_{BFS} is a strongly Nielsen reduced basis.

Theorem 3.4.22 *$\ell(T_{BFS})$ is a strongly Nielsen reduced basis.*

Proof. Let us suppose, by contradiction, that $\ell(T_{BFS})$ is not a strongly Nielsen reduced basis. Then there exist $u, v \in \ell(T_{BFS})$ such that $|uv| < |u|$ or $|uv| < |v|$.
If $|uv| < |v|$ then, letting $u' = u^{-1}$ and $v' = v^{-1}$, one has $|v'u'| = |(uv)^{-1}| = |uv| < |v| = |v'|$. So, we can only consider the case when $|uv| < |u|$.

Let so $|uv| < |u|$. Let $c, d \in B^{\pm}_{T_{BFS}}$ such that $u = l(c)$ and $v = l(d)$. By prop. 3.4.6, there exist p, q, r paths in $\mathcal{A}$ such that $c = pr$, $d = r^{-1}q$, r is contained in T and $|r| > |q|$. In particular r^{-1} and q^{-1} are paths starting at i and ending at the same vertex x. If we consider the paths $\overline{r^{-1}}$ and $\overline{q^{-1}}$ in $\Gamma'_{\mathcal{A}}$ we have that $(\overline{r^{-1}}) > (\overline{q^{-1}})$: in fact $|r| > |q|$ and r and q are reduced.

Moreover, by the proof of prop. 3.1.14, $\overline{r^{-1}}$ is a simple path in T_{BFS} from i to x but it is not the shortest path from i to x in $\Gamma'_{\mathcal{A}}$ since $\overline{q^{-1}}$ is a path from i to x shorter

than the path $\overline{r^{-1}}$. This is a contradiction, by prop. 3.4.21. It follows that $\ell(T_{BFS})$ is a strongly Nielsen reduced basis. □

In the next section we will see an application of this result for finding, given a finite set of words in $F(A)$, an equivalent set (a set generating the same subgroup) that is strongly Nielsen reduced.

Example 3.4.23 *In figure 3.18 we have $\Gamma_{\mathcal{A}}$ the positive state graph of an inverse automaton. The edges underlined are those ones of T_{BFS} the tree obtained applying the BFS procedure in 1 to the undirected graph $\Gamma'_{\mathcal{A}}$. The basis associated to such a tree is*

$$\ell(T_{BFS}) = \{bbab^{-1}, ba^{-1}ba^{-1}, b^{-1}a, a^3b, abb, b^{-1}a^{-1}bab\}$$

Such a basis is a strongly Nielsen basis of $red(L(\mathcal{A}))$. One has that

$$S(T) = 21 < S(T') = 22$$

where the edges of T' are the underlined edges in figure 3.17.

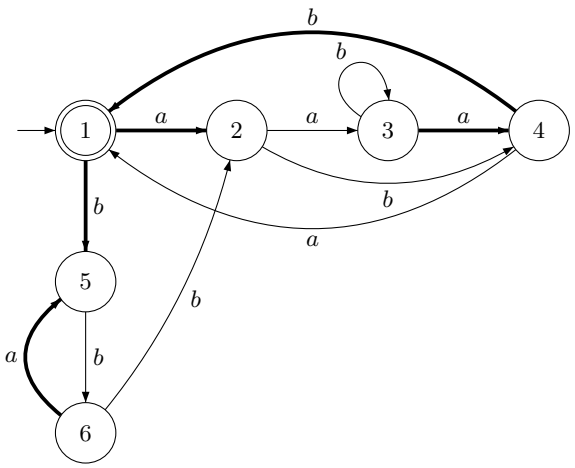

Figure 3.18: $\Gamma_{\mathcal{A}}$ positive state graph of an inverse automaton.

3.4.3 Application: an algorithm for the strongly Nielsen reduction

The Nielsen reduction (see section 1.2) consists of, given a finite set of words U in a free group, finding an equivalent set (i.e. a set generating the same subgroup as U) that is a strongly Nielsen set. The algorithms known in literature work on words (see [14]). The algorithm with the best complexity was found by Avenhaus and Madlener in 1984 (cf. [1]). In particular, they find an algorithm that, given $U = \{u_1, \ldots, u_m\} \subseteq F(A)$ with $|u_i| \leq n$, constructs an equivalent strongly Nielsen set in time $O(m^5 n^2)$. Here we show that, using automata, given U as before, we can construct a strongly Nielsen set equivalent to U in less time than the Avenhaus and Madlener's algorithm.

Let now $U \subseteq F(A)$ be a finite set. Our algorithm consists in constructing the inverse automaton $\mathcal{A}_{<U>}$, considering the BFS tree T_{BFS} of the undirected positive state graph in i and calculating $\ell(B_{T_{BFS}})$, a strongly Nielsen set equivalent to U (i.e. $< \ell(B_{T_{BFS}}) >=< U >$).

ALGORITHM

Input:$U = \{u_1, \ldots, u_m\}$.
Output:V strongly Nielsen reduced such that $< U >=< V >$

1. Construction of $\mathcal{A}_{<U>} = (Q, i, i, \mathcal{F})$
2. Construction of the BFS tree T_{BFS}
3. Construction of the basis associated to T_{BFS}

Let us calculate the complexity of such algorithm for $U = \{u_1, \ldots, u_m\}$ with $k = \sum_{i=1,m} |u_i|$.

1. Time to construct $\mathcal{A}_{<U>}$: $O(k\ log^* k)$

We can construct $\mathcal{A}_{<U>} = (Q, i, i, \mathcal{F})$ in $O(k\ log^* k)$ (cf. [36]) where $log^* : \mathbb{N} \longrightarrow \mathbb{N}$ assigns to each natural i the least natural number j such that $\underbrace{log \cdot log \cdots log}_{j\ times}(i) \leq 1$.
The logarithm is taken in basis 2. In particular, log^* grows so slowly that can be considered a constant.

2. Time to construct the BFS tree T_{BFS}: $O(k)$

We do the breadth-first search (see section 1.4) of $\Gamma'_{\mathcal{A}_{<U>}}$ in i and construct the BFS tree in $O(|Q| + |\mathcal{F}|)$ (see section 1.4). We can suppose $|\mathcal{F}| > |Q|$ since $\Gamma'_{\mathcal{A}_{<U>}}$ is connected. So the BFS tree is constructed in time $O(|\mathcal{F}|)$. While doing the BFS procedure to every vertex x of the graph we associate $\ell(p_x)$, the label in $(A \cup A^{-1})^*$ of

the path in the BFS tree from i to x, $L(x) := \ell(p_x)$. By the construction of $\mathcal{A}_{<U>}$ we have that $|\mathcal{F}| \leq k$. So the BFS tree is build in time $O(k)$.

3. Time to construct $\ell(B_{T_{BFS}})$: $O(k)$

To construct the basis associated to T_{BFS} we have to do another time the BFS search. In such visit to every edge e not in T_{BFS} we associate $L(i(e))\ell(e)L(f(e))^{-1} = \ell(p_{i(e)})\ell(e)\ell(p_{f(e)})^{-1}$. So the basis $\ell(B_{T_{BFS}})$ is build in time $O(k)$.

The total complexity is

$$O(k\ log^* k + 2k) = O(k\ log^* k)$$

Theorem 3.4.24 *The Nielsen reduction for a finite set $U = \{u_1, \ldots, u_m\}$, with $k = \sum_{i=1,m} |u_i|$, can be done in time $O(k\ log^* k)$.*

Remark 3.4.25 *Let now $U = \{u_1, \ldots, u_m\}$, $k = \sum_{i=1,m} |u_i|$ and $|u_i| \leq n$, for each $i = 1, \ldots, m$. One has that $O(m^5 n^2) \geq O(k^2 m^3) > O(k\ log^* k)$.*

By the previous remark, we have that such algorithm has better complexity than that one of Avenhaus and Madlener.

3.5 Pictures

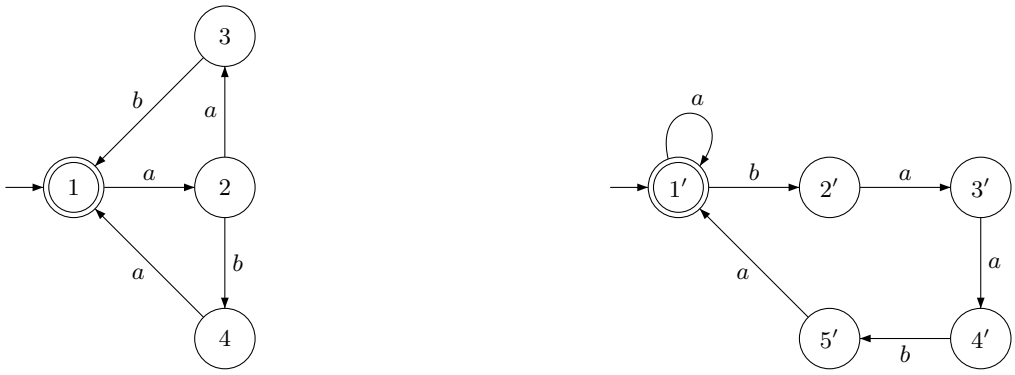

Figure 3.19: $\mathcal{A}_{X^*}$ and $\mathcal{A}_{Y^*}$, with $X = \{aab, aba\}$ and $Y = \{a, baaba\}$. are finitely generated and $X^* \cap Y^* = \{a(abaaba)^*baaba\}^*$ is not finitely generated

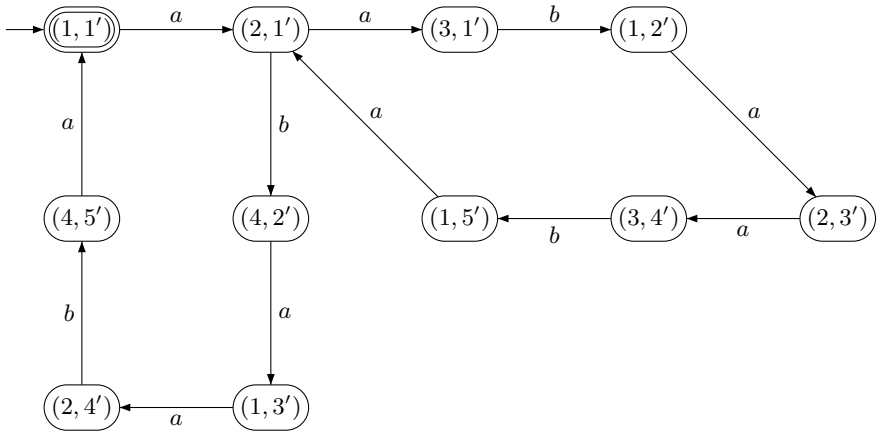

Figure 3.20: $\mathcal{A}_{X^*} \times \mathcal{A}_{Y^*}$ with $X = \{aab, aba\}$ and $Y = \{a, baaba\}$ finitely generated submonoids. One has that $X^* \cap Y^* = \{a(abaaba)^*baaba\}^*$ is not finitely generated

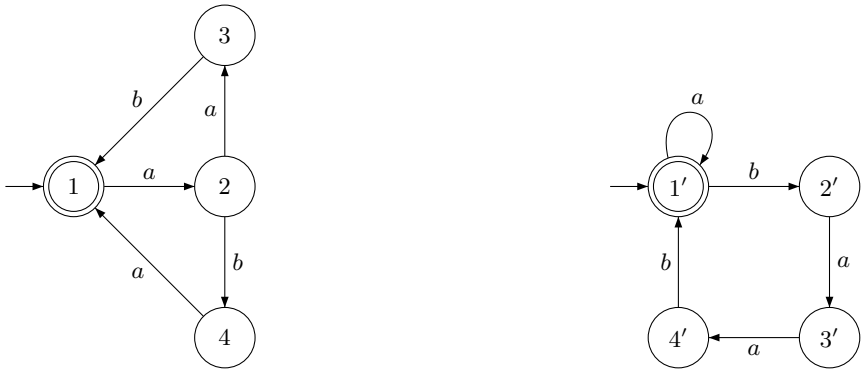

Figure 3.21: $\mathcal{A}_{<X>}$, $\mathcal{A}_{<Y>}$, with $X = \{aab, aba\}$ and $Y = \{a, baaba\}$.

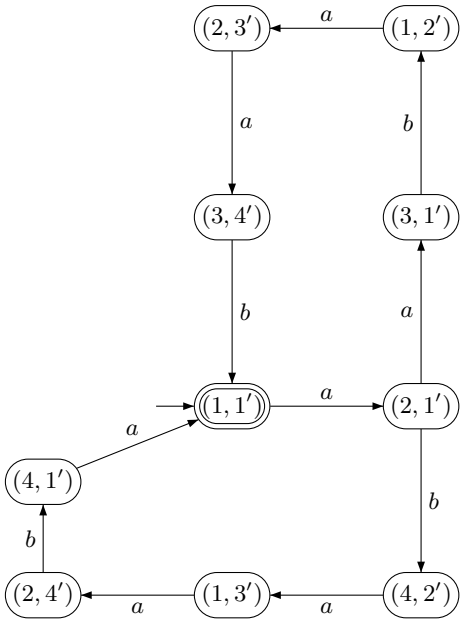

Figure 3.22: $\mathcal{A}_{<X>} \times \mathcal{A}_{<Y>}$ with $X = \{aab, aba\}$ and $Y = \{a, baaba\}$. One has $< X > \cap < Y > = < aabaab, abaaba >$ finitely generated.

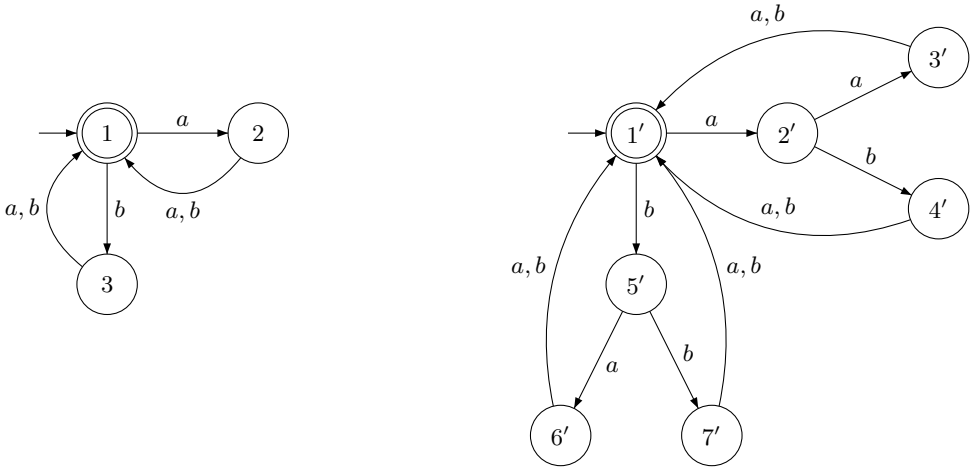

Figure 3.23: $\mathcal{A}_{A^{2*}}$, $\mathcal{A}_{A^{3*}}$ with A^2 the set of words in $A = \{a, b\}$ of length 2 and A^3 the set of words in $A = \{a, b\}$ of length 3

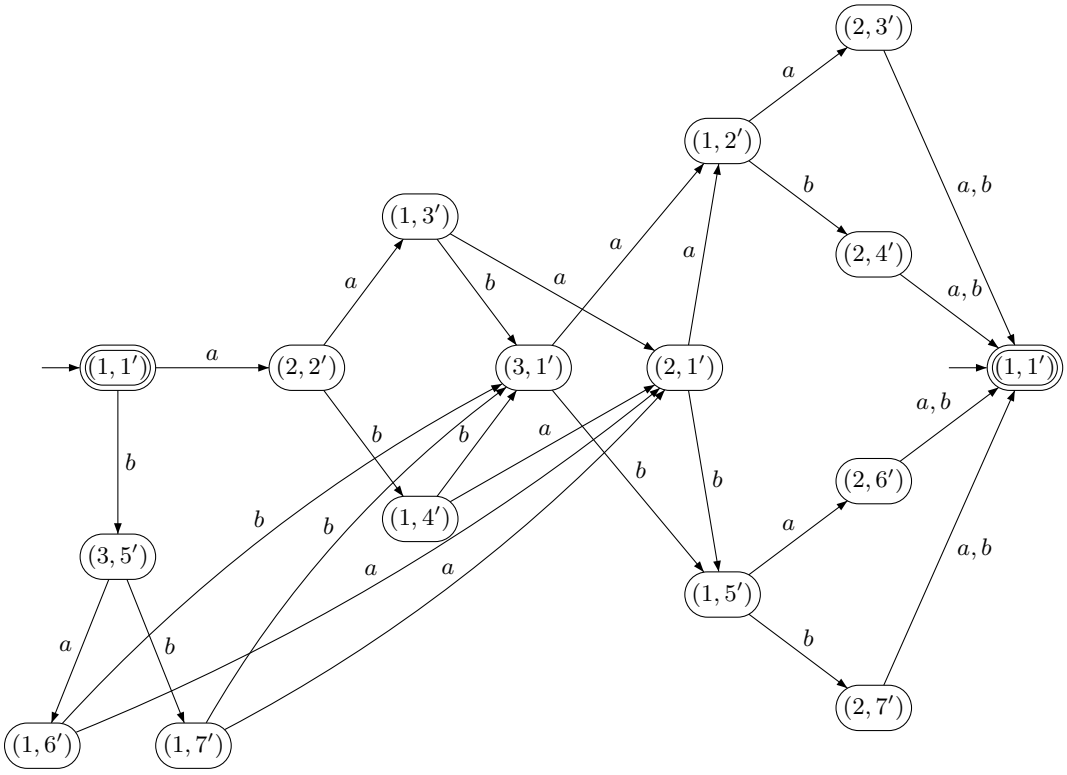

Figure 3.24: $\mathcal{A}_{A^2} \times \mathcal{A}_{A^3}$. This automaton has more than one bpi: $|BPI| = \{5\}$. One has $\widetilde{rk}(H \cap K) = (2^6 - 1) = 64 > \widetilde{rk}(H)\widetilde{rk}(K) = (2^2 - 1)(2^3 - 1) = 21$

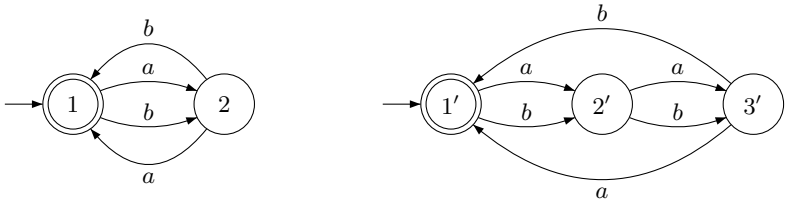

Figure 3.25: $\mathcal{A}_{<A^2>}$, $\mathcal{A}_{<A^3>}$ with A^2 the set of words in $A = \{a, b\}$ of length 2 and A^3 the set of words in $A = \{a, b\}$ of length 3

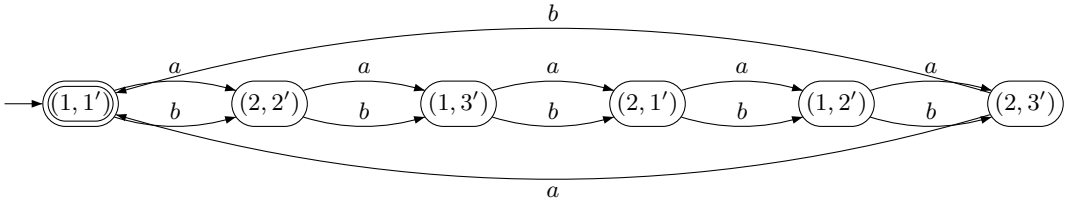

Figure 3.26: $\mathcal{A}_{<A^2>} \times \mathcal{A}_{<A^3>}$ with A^2 the set of words in $A = \{a, b\}$ of length 2 and A^3 the set of words in $A = \{a, b\}$ of length 3

Bibliography

[1] J. Avenhaus and K. Madlener. The Nielsen reduction and P-complete problems in free groups *Theoretical Computer Science*, 32:61–76, 1984.

[2] J. Berstel and D. Perrin. *Theory of Codes*. Academic Press, 1985.

[3] J. Berstel, D. Perrin, J.F. Perrot, A. Restivo. Sur le théorèm du défaut *Journal of Algebra*, 60:169–180, 1979.

[4] J.-C. Birget, S. Margolis, J. Meakin, P.Weil. PSPACE-completeness of certain algorithmic problems on the subgroups of free groups. *Theoretical Computer Science*, 242:247–281, 2000.

[5] V. Bruyére, D. Derencourt, M. Latteux. The meet operation in the lattice of codes. *Theoretical Computer Science*, 191:117–129, 1998.

[6] R. G. Burns. On the intersection of finitely generated subgroups of a free group. *Math. Z.*, 119:121–130, 1971.

[7] J. Clement, J. Duval, G.Guaiana, D. Perrin, G. Rindone. Parsing with a finite dictionary. *Theoretical Computer Science*, 340:432–442, 2005.

[8] W. Dicks and E. Formanek. The rank three case of the Hanna Neumann conjecture. *J. Group Theory*,4,113-151, 2001.

[9] H.Cormen, E. Leiserson, L. Rivest. *Introduction to Algorithms*. The MIT press, 1990.

[10] S. Eilenberg. *Automata, Languages and Machines, Vol. A*. Academic Press, 1974

[11] L. Giambruno, A. Restivo. An automata-theoretic approach to the study of the intersection of two submonoids of a free monoid, *submitted for publication*, 2006.

[12] J.E. Hopcroft, J.D.Ullman *Introduction to Automata Theory, Languages and Computation*. Addison-Weisley Publishing Company, 1979.

[13] A.G. Howson. On the intersection of finitely generated free groups. *J. London Math. Soc:*, 29:428–434, 1954.

[14] R. Lyndon and P.Schupp. *Combinatorial group theory.* Springer, (1977, reprinted 2001).

[15] I. Kapovich, A.G. Miasnikov. Stallings foldings and subgroups of free group. *Journal of Algebra*, 248:2, 608–668, 2002.

[16] J. Karhumäki. A note on intersection of free submonoids of a free monoid. *Semigroup forum*, 29:183–205, 1984.

[17] M. Latteux and J. Leguy. On the composition of morphism and inverse morphisms *Lecture Notes in Computer Science*, 154:420-432, 1983.

[18] M. Lothaire. *Combinatorics on words*, Vol 17 of Encyclopedia of mathematics and its application. Addison-Wesley, 1983.

[19] M. Lothaire. *Algebraic combinatorics on words*, Vol 90 of Encyclopedia of mathematics and its application. Cambridge University Press, 2002.

[20] S. Margolis and J. Meakin. Free inverse monoids and graph immersion. *Intern. J. of Algebra and Computation*, 3:79–100, 1993.

[21] S. Margolis, M. Sapir, P. Weil. Closed subgroups in pro-V topologies and the extension problems for inverse automata. *Intern. J. of Algebra and Computation*, 11:405–445, 2001.

[22] J.Meakin and P.Weil. Subgroups of free groups: a contribution to the Hanna Neumann conjecture. *Geometriae Dedicata*, 94:33–43, 2002.

[23] H. Neumann. On intersections of finitely generated subgroups of free groups. *Publ. Math.Debrecen*, 4:186–189, 1956.

[24] W.D. Neumann. On intersections of finitely generated subgroups of free groups. *Lect. Notes in Math.*, 1456:161–170, 1990.

[25] Reidmeister. Fundamentalgruppen and Überlagerungsräume. *Nachrichten der Ges. Wiss., math. Phys. Klasse Göttingen*, 69–76, 1928.

[26] D. Perrin and G. Rindone. Syntactic groups. *Bullettin o fthe Belgium Mathematical Society*,10:749-759, 2003

[27] J. Sakarovitch. *Elements de theorie des automates.* Vuibert Informatique, 2003.

[28] M.P. Schützenberger. Une theorie algebrique du codage. in *Seminarie Dubreil-Pisot*, annèe 1955-56, Institut H. Poincaré, Exposé N.15.

[29] M.P. Schützenberger. A property of finitely generated submonoids of free monoids. *Algebraic theory of semigroups (Proc. Sixty Algebraic Conf., Szeged, 1976)*, 545–576, North-Holland, Amsterdam,1979.

[30] J.-P. Serre. Arbres, amalgames, SL_2. *Asterisque*, 1456:161–170, 1990.

[31] P. Silva and P.Weil. On an algorithm to decide whether a free group is a free factor of another. *Theoretical Computer Science*, to appear.

[32] J.R. Stallings. Topology of finite graphs. *Inventiones Math.*, 71:551–565, 1983.

[33] G. Tardos. On the intersection of subgroups of a free group. *Invent. Math*, 108:29–36,1992

[34] G. Tardos. Towards the Hanna Neumann conjecture using Dicks' method. *Invent. Math*, 123:95–104,1996

[35] B. Tilson. The intersection of free submonoids of a free monoid is free. *Semigroup forum*, 4:345–350, 1972.

[36] N. Touikan. A fast algorithm for Stalling's folding process, *preprint*, 2005.

[37] P.Weil. Computing closures of finitely generated subgroups of the free group, in in *Algorithmic problems in groups and semigroups*(J.-C. Birget, S. Margolis, J. Meakin, M. Sapir éds.), Birkhaüser, 289–307, 2000.