The subgroup membership problem

Andrew J. Duncan, Elizaveta Frenkel

November 28, 2012

Abstract

Stallings folding theory is modified, using double coset representatives, and to applied to the study of subgroups of amalgamated products of finite rank free groups. As an application the subgroup membership problem for such groups is shown to be decidable. An algorithm for this problem is constructed and its complexity is analysed.

1 Introduction

se:global_intro

Stallings introduced his technique of subgroup foldings in [18] where he applied it to investigate subgroups of free groups. In [17] Stallings extended ideas of foldings to non-free actions of groups on graphs and trees. In the following years these ideas were widely generalised, in several directions, and used to solve a variety of problems in combinatorial group theory: see for example [2, 7, 8, 16]. In this paper we consider subgroups of free products with amalgamation of free groups of finite rank, using a language of normal forms based on double coset representatives. We extend the theory of Stallings foldings to this setting and use it to construct an explicit algorithm for the subgroup membership problem.

Let $G =< X|R >$ be a finitely generated group. The *subgroup membership problem*, for fixed subgroup K of G, is to decide whether or not a given word w in $\mathbb{F}(X)$ represents an element of the subgroup K. The *uniform* subgroup membership problem for G is to decide, given a finite subset $P = \{w, k_1, \dots, k_s\}$ of elements of $\mathbb{F}(X)$, for some $s \geq 0$, whether or not w represents an element of the subgroup K of G generated by $Y = P\backslash\{w\}$. The subgroup membership problem for K in G is said to be *solvable* if there is an algorithm which, on input $w \in \mathbb{F}(X)$, outputs "yes" if and only if w does represent an element of K. Similarly, the uniform subgroup membership problem for G is *solvable* if there exists an algorithm which, on input P, outputs "yes" if and only if w represents an element of K.

Associated to the subgroup membership problem for the subgroup K of the group G, generated by a finite set $Y = \{k_1, \ldots, k_s\}$, as above, is the *subgroup membership search problem*: given an element $w \in \mathbb{F}(X)$ such that w represents an element of K, find a sequence of pairs (k_{i_j}, ε_j), $m = 1, \ldots, t$, for some $t \in \mathbb{N}$, $k_{i_j} \in \{k_1, \ldots, k_s\}$ and $\varepsilon_j = \pm 1$, such that $w = k_{i_1}^{\varepsilon_1} \cdots k_{i_t}^{\varepsilon_t}$ in the group G. The subgroup K is said to have *solvable membership search problem* if there is an algorithm which, on input a word w in $\mathbb{F}(X)$ representing an element of K, will output such a sequence. The *uniform membership search problem* and its solvability, for a group G, are defined in the obvious way.

If $\mathcal{V}$ is a class of groups and there exists an algorithm to solve the uniform membership (search) problem for any element of $\mathcal{V}$ then we say that $\mathcal{V}$ has solvable uniform (search) membership problem.

thm:membership **Theorem 1.1.** *Let F_1 and F_2 be finite rank free groups and let H_1 and H_2 be finitely generated subgroups of F_1 and F_2, respectively, such that H_1 is isomorphic to H_2. Let $G = F_1 *_{H_1=H_2} F_2$. Let $\mathcal{F}$ be the class of free products with amalgamation, where factors are free of finite rank, and the amalgamated subgroups are finitely generated. Then*

it:membership 1. *G has solvable membership problem,*

t:uni-membership 2. *G has solvable uniform membership problem and*

s-uni-membership 3. *$\mathcal{F}$ has solvable uniform membership problem.*

Moreover the same conclusions hold for the search versions of each of these problems.

"Free group constructions", in particular, free products of groups, with or without amalgamation, HNN-extensions and, more generally, fundamental groups of graphs of groups constitute a subject of special significance in group theory, from many points of view. Initial results in this area, related to the solvability of membership problem, due to Mihailova (see [14, 15]), show that the membership problem is decidable in the free product of groups A and B if it is decidable in both factors. Another famous result of Mihailova [13] provides an important counter-example: namely, a finitely generated subgroup of a direct product of two free groups of rank two with unsolvable membership problem. This direct product can in fact be considered as a sequence of two HNN-extensions of rank two free group. On the other hand, another classical result [1, Proposition 2.21] shows that a free group of finite rank has solvable (uniform) membership problem. Thus, even innocent group constructions can dramatically affect solvability of the membership problem.

The membership problem for free products with amalgamation and HNN-extensions was studied by Bezverkhnii in papers [3, 4, 5, 6], using combinatorial techniques and standard normal forms for amalgamated products of groups. Our approach is, first of all, more geometric, and since we have an eye on a generalisation of results to other free constructions, we have been led to introduce alternative normal forms appropriate to study of these constructions.

Another significant step in development of the theory of foldings for free constructions was taken by Kapovich, Miasnikov and Weidmann [9], where the authors investigated the uniform membership problem for fundamental groups of graphs of groups, under certain assumptions on both vertex and edge groups. In particular, they showed that, in the case where all vertex groups are either locally quasiconvex word hyperbolic or polycyclic-by-finite, and all edge groups are polycyclic-by-finite; the uniform membership problem for the fundamental group of the graph of groups is solvable.

As mentioned above, we introduce a geometric technique, which relies on Stallings foldings and normal forms based on double cosets, for elements of amalgamated products of groups.

Now, a few words on the structure of the paper. In Section 2 we introduce double coset repreesentatives for elements of free groups, with respect to a fixed subgroup, and describe an explicit procedure for their construction. These representatives are used to define unique double coset normal forms for elements of amalgamated products of finite rank free groups in Section 3, which is the heart of the paper. For a subgroup K of such an amalgam, we describe a generalised folding process, which begins with the flower automaton of K, over an extended alphabet, and results in a "double coset graph" for K, which recognises precisely the normal forms of elements of K.

In Section 6 we estimate the time complexity of algorithms described in the previous parts of the paper. In particular, we illustrate how quickly the preparatory work in construction of double coset normal forms of generators for K can be carried out; we calculate the complexity of the process of modification of components of the flower automaton of K, and of the reassembly process required to produce the double coset graph.

The authors are grateful to Vladimir Remeslennikov for suggesting the use of double cosets and for helpful discussions, and to Christian Perfect who gave invaluable help with programming.

2 Constructing double coset representatives

sec:dcforms

2.1 Free products with amalgamation

sec:intro

We denote the free group on as set X by $\mathbb{F}(X)$. By a *reduced* word in $\mathbb{F}(X)$ we mean a freely reduced word in $(X \cup X^{-1})^*$, and we write $w \in \mathbb{F}(X)$ to mean that w is a reduced word. For $u, v, w \in \mathbb{F}(X)$ we write $w = u \circ v$ if $|w| = |u| + |v|$, in which case we say that u is a *prefix* of w.

Let X_1 and X_2 be disjoint, finite sets and let $F_1 = \mathbb{F}(X_1)$ and $F_2 = \mathbb{F}(X_2)$. Let $H_1 \leq F_1$ and $H_2 \leq F_2$ be finitely generated subgroups of rank m, freely generated by $\{h_1, \ldots, h_m\}$ and $\{h'_1, \ldots, h'_m\}$, respectively. Denote by ϕ the isomorphism of H_1 to H_2 which maps h_i to h'_i, for all i. Now let $G = F_1 \underset{H_1=H_2}{*} F_2$ be the free product of F_1 and F_2, amalgamating H_1 and H_2: that is G is the group with presentation

$$\langle X_1 \cup X_2 | h_i = h'_i, i = 1, \ldots, m\rangle.$$

Given an arbitrary word $g \in F_1 * F_2$, we should like an algorithm to write g in some normal form with uniqueness. As a first attempt, we say g is in *reduced form* if either g is the empty word 1, or $g = g_1 \cdots g_t$, where

- $g_1 \in F_1$ or $g_1 \in F_2$ and, if $t = 1$ then $g_1 \neq 1$, and
- for $i > 1$, $g_i \in (F_1 \backslash H_1) \cup (F_2 \backslash H_2)$ and
- for $i \geq 1$, g_i and g_{i+1} belong to different factors.

As the membership problem is solvable in F_1 and F_2 we may write elements in reduced form: for example, to write g in reduced form, we can use the following procedure. First write g as a reduced word in $F_1 * F_2$, so $g = f_1 \cdots f_t$, where each f_i belongs to a factor and consecutive f_i come from different factors, F_1 or F_2. Then apply the following process.

it:st1 Step 1 For $i = 1, \ldots, t$ do the following. Write each f_i as a reduced word in F_1 or F_2 as appropriate. If $f_i \in F_k$ then, using an algorithm for the membership problem in H_k, check whether or not $f_i \in H_k$. If none of the f_i's lies in H_1 or H_2 then $g = f_1 \cdots f_t$ is in reduced form.

Step 2 Now suppose that some $f_i \in H_k$, and let i be the minimal index with this property. Then $\phi^{\pm 1}(f_i) = c \in H_l$, $l \neq k$. If $i = 1$, rewrite g in the form $g = f'_2 f_3 \cdots f_t$, where $f'_2 = c f_2$, an element of $F_1 \cup F_2$. If $i > 1$ then rewrite g as $g = f_1 \cdots f_{i-2} f'_{i+1} f_{i+2} \cdots f_t$, where $f'_{i+1} = f_{i-1} c f_{i+1} \in F_1 \cup F_2$. Return to it:st1 Step 1.

The reduced form is easy to calculate and gives a solution to the word problem: since a reduced form with $t > 0$ cannot represent the identity [11, Theorem 2.6]. However reduced forms do not uniquely represent elements of G. To resolve this non-uniqueness problem it is common to chose a set of coset representatives of H_k in F_k, $k = 1, 2$, and use these to rewrite the reduced form. If T_k is a set of right coset representatives of H_k then a word $g \in F_1 * F_2$ is in *single coset normal form* (or *sc-normal form*) if

$$g = ct_1 \cdots t_r,$$

where $c \in H_1$, $t_i \in T_1 \cup T_2$, $t_1 \neq 1$, and consecutive t_i are from different factors. Every element of G can be *uniquely* expressed as a word in single coset normal form [12, Theorem 4.4]. Coset representatives can be found using Stallings automata, which we describe below, so there is a procedure for writing elements in sc-normal form. Namely: having written an element g in reduced form, working from right to left, for each i we express f_i in the form $c_i t_i$, where $c_i \in H_k$ and $t_i \in T_k$. The element f_{i-1} must then be replaced by $f_{i-1}\phi^{\pm 1}(c_i)$, before being rewritten. Continue this way till the left hand side of the word is reached.

In this process the coset representative of f_{i-1} with respect to H_l will not, in general, be known until f_i, ..., f_t have been rewritten, and moreover each step requires an application of the map ϕ or its inverse. As we shall see, when we write words in terms of double coset normal forms the algorithm is simpler, and the coset representatives that occur in any reduced form of a word are the same as the double coset representatives of the final double coset normal form.

To write elements in terms of single or double coset representatives we use Stallings automata, which we now define. In the following subsection we shall then define double coset representatives.

2.2 Automata and Stallings Foldings

sub:foldings

An *automaton* A is a quintuple $(\Sigma, Q, \delta, \mathcal{S}, \mathcal{F})$, where Σ is finite set called the *alphabet*, Q is a finite set of *states*, δ is a map $\delta : Q \times \Sigma \to Q$, called the *transition function*, $\mathcal{S} \subset Q$ is the (non-empty) set of *start states* and $\mathcal{F} \subseteq Q$ is the set of *final states*. If $\mathcal{F} = \{s_0\} = \mathcal{S}$ it's common to drop $\mathcal{F}$ from the description and define A as a quadruple. For details of the theory of automata the reader is referred to [10].

By a graph we mean a finite, directed, edge labelled graph. We shall associate to an automaton A a graph Γ_A with vertices $V = V(\Gamma_A) = Q$; edge set $E = E(\Gamma_A)$ consisting of elements (u, σ, v) of $Q \times \Sigma \times Q$ such that

$\delta(u,\sigma) = v$; and labelling function $l : E \to \Sigma$ given by $l(u,\sigma,v) = \sigma$, for all $(u,\sigma,v) \in E$. If $\mathcal{S}$ and $\mathcal{F}$ are the sets of start and final states of A we sometimes write $(\Gamma_A, \mathcal{S}, \mathcal{F})$ for $(\Sigma, Q, \delta, \mathcal{S}, \mathcal{F})$, and in addition if $\mathcal{S} = \{s_0\}$ and $\mathcal{F} = \{f_0\}$, we may abbreviate this to (Γ_A, s_0, f_0). If $\mathcal{S} = \{s_0\} = \mathcal{F}$ we call s_0 the *root* vertex of Γ_A and say that A and Γ_A are *rooted*. We may write (Γ_A, s_0) for (Γ_A, s_0, f_0) if this is the case. For notational simplicity we identify Γ_A and A whenever it is convenient and does no ambiguity arises.

By a path p in a graph we mean a sequence $(v_0, e_1, v_1, \ldots, e_n, v_n)$ of vertices v_i and edges e_i such that $e_i = (v_{i-1}, \sigma_i, v_i)$, for $i = 1, \ldots, n$. The label $l(p)$ of this path is defined to be the concatenation $\sigma_1 \cdots \sigma_n$ of the labels of the edge sequence of p. If w is the label of a path in Γ_A, from a vertex of $\mathcal{S}$ to a vertex of $\mathcal{F}$, then we say that w is accepted by $A = (\Gamma_A, \mathcal{S}, \mathcal{F})$. The set of all words in the free monoid Σ^*, generated by Σ, which are accepted by A is called the *language accepted by* A and denoted $L = L(A)$. We extend this terminology, for an automaton $A = (\Sigma, Q, \delta, \mathcal{S}, \mathcal{F})$, to say that a word w is *readable* by A if w is in the language accepted by $(\Gamma_A, \mathcal{S}, Q)$ and define

$$L_Q = L_Q(A) = \{w : w \text{ is readable by } A\}.$$

If Σ is a subset of a group G (and whenever $a, a^{-1} \in \Sigma$ then a^{-1} is the inverse of a) then the natural map from $L_Q(A)$ to G is denoted $\pi : L_Q(A) \to G$.

An *involutive alphabet* is a set Σ of the form $\{a_1, \ldots, a_r, a_1^{-1}, \ldots, a_r^{-1}\}$. The automaton $A = (\Sigma, Q, \delta, \mathcal{S}, \mathcal{F}) = (\Gamma_A, \mathcal{S}, \mathcal{F})$ is *involutive* if Σ is an involutive alphabet and, for all $p, q \in Q$ and $a \in \Sigma$, $\delta(p,a) = q$ if and only if $\delta(q, a^{-1}) = p$ (that is $(u,a,v) \in E \Leftrightarrow (v, a^{-1}, u) \in E$). In diagrams of involutive automata an edge labelled a from u to v represents both (u,a,v) and (u, a^{-1}, v).

From now on we will consider only involutive automata and, moreover, we always assume that Σ is a subset of a group. The automaton A is *deterministic* or *folded* if $\delta(p,a) = q$ and $\delta(p,a) = q'$ imply $q = q'$ (that is, $(u,a,v) \in E$ and $(u,a,v') \in E$ imply $v = v'$). We say an automaton A is *connected* if Γ_A is connected. (For an involutive rooted automataton being connected is equivalent to the property that for every vertex q there is a path from q to the root and a path from the root to q. Automata with this property are usually called *trim*, but since the two properties are equivalent in our setting we use the more intuitive nomenclature.) Finally, an involutive, rooted, connected, deterministic, automaton, with a single final state, is called an *inverse* automaton.

Let $A = (\Gamma_A, s_0, f_0)$ be an inverse automaton. If w is readable by A then there exists a unique vertex $\tau(w)$ of Γ_A such that w is accepted by $(\Gamma_A, s_0, \tau(w))$: we call $\tau(w)$ the *terminal vertex* of w. (L is the set of words

w readable by A with $\tau(w) = s_0$.) Fix a spanning tree T for Γ_A. For each vertex v of Γ_A let $w(v)$ denote the label of the path in T from s_0 to v. Define

$$L_T = L_T(A) = \{w(v) : v \text{ is a vertex of } A\}.$$

Thus $L \subseteq L_T \subseteq L_Q$.

If $w, s, u \in \mathbb{F}(X)$ with $w = s \circ u$ then we say that s is

(i) an *L_Q-prefix* of w if $s \in L_Q$;

(ii) an *L_T-prefix* of w if $s \in L_T$.

An L_Q or L_T-prefix s of w is *maximal* if no longer subword of w is an L_Q or L_T-prefix, respectively.

Since we identify the automaton A with the graph Γ_A, we ascribe properties of automata (like rooted, connected, involutive, folded or inverse) to graphs in general and Γ_A in particular.

We now recall the notion of a Stallings folding of an automaton, which we will use directly, to construct Stallings automata for subgroups of free groups, and in generalised form to produce dc-resolutions of automata (in Section 3). For more details on Stallings foldings and Stallings automata see [20] or [1].

An *elementary folding* of a rooted, directed, labelled graph (Γ, s_0) is a graph (Γ', s_0') obtained from (Γ, s_0) as follows. Suppose that $e = (u, \sigma, v)$ and $e' = (u, \sigma, v')$ are edges of Γ. (We do not require u, v and v' to be distinct.) Then Γ' is the quotient of Γ formed by identifying v and v', to form a new vertex v''; and e and e', to form a new edge $e'' = (u, \sigma, v'')$. If $s_0 = v$ or $s_0 = v'$ then v'' is the root of Γ' and otherwise $s_0' = s_0$. A *folding* of a graph Γ is a graph obtained from Γ by a finite sequence of elementary foldings.

If a rooted graph Γ is not folded then a folding may be applied; reducing the number of edges of the graph. Continuing this way a folded graph may eventually be produced. A folding of Γ which is folded is called a *Stallings folding of* Γ. It follows that folded (i.e. deterministic) graphs are precisely those to which no elementary folding may be applied.

A morphism of automata Γ_A and $\Gamma_{A'}$ is a map $\theta : \Gamma_A \to \Gamma_{A'}$ which maps vertices to vertices and edges to edges in such a way that

1. if s is a start state of Γ_A then $\theta(s)$ is a start state of Γ';

2. if t is a final state of Γ then $\theta(t)$ is a final state of Γ' and

3. if (u, σ, v) is an edge of Γ then $\theta(u, \sigma, v) = (\theta(u), \sigma, \theta(v))$.

An elementary folding of (Γ, s_0, t) to (Γ', s_0', t') induces a morphism from Γ to Γ': namely the quotient map. Therefore there is a uniquely determined *folding morphism* from Γ to any folding. Moreover, if Γ_1 is a folding of Γ and θ is the folding morphism then $\pi(L(\Gamma, s, t)) = \pi(L(\Gamma_1, \theta(s), \theta(t)))$.

Let $Y = \{w_1, \ldots, w_n\}$ be a generating set for H, where $w_i \in F$. The *flower automaton* $\Gamma_Y(H)$, of H (with respect to this generating set) is the graph constructed as follows. For each i let C_i be a cycle graph with $|w_i|$ vertices, and choose a vertex v_i as the root. Direct and label the edges of C_i, with elements of X, so that the simple closed path based at v_i has label w_i (read in, say, a counter-clockwise direction). The flower automaton $G_Y(H)$ is formed by identifying all the vertices v_i to form a new graph, with root v the image of the v_i. If L_Y is the language accepted by $(\Gamma_Y(H), v)$ then $\pi(L_Y) = H$.

The flower automaton of H depends on the chosen generating set, and is, in general, non-deterministic (at the root vertex s). Stallings stallings83 [18] proved that the folded graph $\Gamma(H)$, obtained by applying foldings to the flower automaton of H until the result is folded, is an inverse automaton, independent of the generating set chosen. Moreover $(\Gamma(H), s)$ is the minimal (fewest states) automaton accepting every reduced word representing an element of H. (See BartholdiSilva [1] for a proof in terms of automata.)

Definition 2.1. *The automaton* $(\Gamma(H), s)$ *is called the* Stallings automaton *of* H.

sub:2cosetrepr

2.3 Double coset representatives in free groups

Let X be a finite alphabet, $F = \mathbb{F}(X)$ and H a finitely generated subgroup of F. A set $S \subseteq F$ such that

1. $F = \bigcup_{s \in S} HsH$ and

2. for all $s, s' \in S$, $s \in Hs'H$ implies $s = s'$,

is called a set of *double coset representatives for* $H \leq F$.

Let A be the Stallings automaton for H, and let $s_0 = 1$ be its start state. We shall define a set of double coset representatives for H in two parts. We begin with the definition of the first type of representative.

def:repres_t1

Definition 2.2 (Double coset representative, type 1)**.** *A word* $w \in \mathbb{F}(X)$ *is a* (double coset) representative of type 1 *if*

$$w = s \circ e \circ t^{-1},$$

where $e \neq 1$, s is a maximal L_Q-prefix and an L_T-prefix of w, and t is a maximal L_Q-prefix and an L_T-prefix of $t \circ e^{-1}$. Let $S^{(1)}$ denote the set of all representatives of type 1.

To describe the remaining representatives we shall first define an equivalence relation on the ordered pairs of distinct vertices of A. Let

$$P = \{(u, v) \in V(A) \times V(A) : u \neq v\}.$$

Define a relation $\sim$ on P by $(u_0, u_1) \sim (v_0, v_1)$ if and only if there exist paths p_0 and p_1 in A, from u_0 to v_0 and u_1 to v_1 respectively, such that $l(p_0) = l(p_1)$. (We allow these paths to have length 0.) Then $\sim$ is an equivalence relation on P.

lem:equiv_verts **Lemma 2.3.** *Let (u_0, u_1) and (v_0, v_1) be elements of P and let $a_0 = w(u_0)$, $a_1 = w(u_1)$, $b_0 = w(v_0)$ and $b_1 = w(v_1)$. Then $(u_0, u_1) \sim (v_0, v_1)$ if and only if there exist $h_0, h_1 \in H$ such that*

$$a_0 a_1^{-1} = h_0 b_0 b_1^{-1} h_1^{-1}.$$

Proof. $\Rightarrow$: Let p_0 and p_1 be paths, from u_0 to v_0 and u_1 to v_1 respectively, such that $l(p_0) = l(p_1) = c$, say. Set $h_0 = a_0 c b_0^{-1}$ and $h_1 = a_1 c b_1^{-1}$. Since h_0 and h_1 are labels of closed paths in A, based at 1, we have h_0 and h_1 in H. Thus $a_0 a_1^{-1} = h_0 b_0 c^{-1} c b_1^{-1} h_1^{-1} = h_0 b_0 b_1^{-1} h_1^{-1}$, as required.

$\Leftarrow$: Let h_0, $h_1 \in H$ such that $a_0 a_1^{-1} = h_0 b_0 b_1^{-1} h_1^{-1}$. Set $k = a_0^{-1} h_0 b_0 = a_1^{-1} h_1 b_1$. Then $h_0 = a_0 k b_0^{-1}$ and $h_1 = a_1 k b_1^{-1}$ belong to H so there exist paths p_0 and p_1 in A, from $\tau(a_0) = u_0$ to $\tau(b_0) = v_0$ and $\tau(a_1) = u_1$ to $\tau(b_1) = v_1$, both with labels k. Therefore $(u_0, u_1) \sim (v_0, v_1)$. □

To work with the equivalence relation $\sim$ its useful to define the product of graphs. Given (labelled, directed) graphs Γ_1 and Γ_2 we define the *product* graph $\Gamma_1 \times \Gamma_2$ to be the graph with vertices $V = V(\Gamma_1) \times V(\Gamma_2)$ and with a directed edge labelled a from (u_1, u_2) to (v_1, v_2) if and only if there are edges (u_1, a, v_1) and (u_2, a, v_2) in Γ_1 and Γ_2, respectively. If Γ is a graph then vertices of $\Gamma \times \Gamma$ of the form (v, v) are called *diagonal vertices*. Note that if Γ is folded there is no edge of $\Gamma \times \Gamma$ joining a diagonal vertex to a non-diagonal vertex. In this case it follows that the diagonal vertices are the vertices of a connected component of $\Gamma \times \Gamma$, which is isomorphic to Γ, and that no other connected component contains a diagonal vertex.

In this notation P is the set of non-diagonal vertices of $\Gamma_A \times \Gamma_A$ and $(u, v) \sim (u', v')$ if and only if there is a path from (u, v) to (u', v') in $\Gamma_A \times \Gamma_A$. Thus, as Γ_A is folded, the equivalence class of $(u, v) \in P$ is the set of vertices of the connected component of $\Gamma_A \times \Gamma_A$ containing (u, v).

We shall choose one double coset representative corresponding to each $\sim$ equivalence class. First observe that if $(u, v) \in P$ and $w(u) = a \circ x$, $w(v) = b \circ x$, for some $a, b \in \mathbb{F}(X)$ and $x \in X^{\pm 1}$, then $(\tau(a), \tau(b)) \in P$ and $(u, v) \sim (\tau(a), \tau(b))$. It follows that every equivalence class of $\sim$ contains an element (u', v') such that $w(u')w(v')^{-1}$ is a reduced word; and representatives of each equivalence class will be chosen to have this property.

def:repres_t2 **Definition 2.4.** *Let* $\mathbf{p}$ *be an equivalence class of* $\sim$ *and let* Y *be the set of all pairs* $(u, v) \in \mathbf{p}$ *such that* $|w(v)|$ *is minimal (amongst elements of* $\mathbf{p}$*). Choose* $(u, v) \in Y$ *such that* $|w(u)|$ *is minimal (amongst elements of* Y*) and define* (u, v) *to be the* $\sim$ representative *of* $\mathbf{p}$*. Let* P_0 *denote the set of all these* $\sim$ *representatives.*

A word $w \in \mathbb{F}(X)$ *is a* (double coset) representative of type 2 *if* $w = w(u)w(v)^{-1}$*, for some* $(u, v) \in P_0$*. Let* $S^{(2)}$ *denote the set of all representatives of type* 2*.*

For each (u, v) *in* P *choose a path in* $\Gamma_A \times \Gamma_A$ *from* (u, v) *to the* $\sim$ *representative* (u_0, v_0) *of* (u, v) *and define the* connecting element $c(u, v)$ *of* (u, v) *to be the label of this path.*

The connecting elements enable systematic rewriting of certain elements of $\mathbb{F}$ in terms of representatives of type 2, as will be seen below.

Definition 2.5. *Define* $S = S^{(1)} \cup S^{(2)}$.

prop:dcreps **Proposition 2.6.** *S is a set of double coset representatives for H.*

Proof. First we shall show that every element $w \in \mathbb{F}(X)$ lies in HdH, for some $d \in S$. In fact we describe an algorithm which rewrites a given word w in this form.

Assume that the Stallings automaton A for H has been constructed by folding from given generators for H. A spanning tree T for A may then be chosen and the set L_T computed. To find the equivalence class of a pair $(u, v) \in P$ the graph $\Gamma_A \times \Gamma_A$ is constructed. As we pointed out above, $(u', v') \sim (u, v)$ if and only if there is a path in $\Gamma_A \times \Gamma_A$ from (u', v') to (u, v). Therefore the $\sim$ equivalence class of (u, v) is the vertex set of the connected component of $\Gamma_A \times \Gamma_A$ containing (u, v). Having constructed the equivalence classes of $\sim$, a set of $\sim$ representatives may be constructed, by considering the lengths of $w(a)$ and $w(b)$ for all (a, b) in an equivalence class. Thus the set P_0 and the connecting element $c(u, v)$, of each element $(u, v) \in P$, may be computed at this stage.

Algorithm I.
Input $w \in \mathbb{F}(X)$. Let h be the maximal prefix of w accepted by A; so $h \in H$

and $w = h \circ f$, for some $f \in \mathbb{F}(X)$. Now use A to find the maximal L_Q-prefix p of f (i.e. the maximal prefix readable by A). Then $f = p \circ q$, for some $q \in \mathbb{F}(X)$.

Next find the maximal prefix g of q^{-1} acceptable by A: say $q^{-1} = g \circ r$, for some $r \in \mathbb{F}(X)$, and then the maximal L_Q-prefix t of r; say $r = t \circ e^{-1}$, for some $e \in \mathbb{F}(X)$.

If $e \neq 1$ then

$$w = h \circ p \circ e \circ t^{-1} \circ g^{-1},$$

with $h, g \in H$. In this case p and t are readable by A: so are L_Q-maximal subwords of f and r, respectively, but may not be in L_T. Hence the next step is to replace p and t by products of the form uv, where $u \in H$ and v is in L_T, if necessary. Set $y = w(\tau(p))$ and $z = w(\tau(t))$. (If p is in L_T then we merely set $y = p$; and similarly if t is in L_T then $z = t$.) Then py^{-1} and tz^{-1} belong to H. Moreover, as p is the maximal L_Q-prefix of f it is also the maximal L_Q-prefix of pet^{-1}, and so the first letter of e is not readable from the vertex $\tau(p) = \tau(y)$ of A. In particular $ye = y \circ e$. Similarly, the first letter of e^{-1} is not readable from the vertex $\tau(t) = \tau(z)$, so $ez^{-1} = e \circ z^{-1}$. Moreover y is both a maximal L_Q-prefix and an L_T-prefix of yez^{-1} and z is a maximal L_Q-prefix and an L_T-prefix of ze^{-1}. Thus $yez^{-1} \in S^{(1)}$ and we output

$$w = (hpy^{-1})(yez^{-1})(zt^{-1}g^{-1}) \in HSH,$$

of the required form.

On the other hand if $e = 1$ then

$$w = h \circ p \circ t^{-1} \circ g^{-1}.$$

In this case let $u = \tau(p)$ and $v = \tau(t)$, let (u_0, v_0) be the $\sim$ representative of the equivalence class of (u, v), $c = c(u, v)$, $y = w(u_0)$ and $z = w(v_0)$. Then, as in the proof of Lemma 2.3, setting $h_0 = w(u)cy^{-1}$ and $h_1 = w(v)cz^{-1}$ we have $h_0, h_1 \in H$ and

$$w(u)w(v)^{-1} = h_0 y z^{-1} h_1^{-1}.$$

Furthermore $pw(u)^{-1}$ and $tw(v)^{-1}$ are in H. Hence

$$p \circ t^{-1} = (pw(u)^{-1})w(u)w(v)^{-1}(w(v)t^{-1}) = (pw(u)^{-1})h_0 y z^{-1} h_1^{-1}(w(v)t^{-1})$$

and by definition $yz^{-1} \in S^{(2)}$. We then output

$$w = ayz^{-1}b,$$

where $a = h(pw(u)^{-1})h_0 = hpcy^{-1} \in H$ and $b = h_1^{-1}(w(v)t^{-1})g^{-1} = zc^{-1}t^{-1}g^{-1} \in H$.

It remains to show that if $s_0, s_1 \in S$ with $s_1 \in Hs_0H$ then $s_1 = s_0$. Suppose that $s_1 = as_0b$, where $a, b \in H$. If $s_0 \in S^{(1)}$, say $s_0 = y_0e_0z_0^{-1}$, then, as ay_0 and $b^{-1}z_0$ are readable by A and $e_0 \neq 1$, it follows (as above) that s_1 cannot be factored as $s_1 = yz^{-1}$, where both y and z are readable by A. Hence both s_0 and s_1 belong to $S^{(1)}$ or both belong to $S^{(2)}$.

Consider the case where both belong to $S^{(1)}$ and, in the usual notation, $s_i = y_ie_iz_i^{-1}$, $i = 0, 1$. We have $s_1 = as_0b$, $a, b \in H$, and as y_0 has no left divisor in H, if $a \neq 1$ then a does not cancel completely with a prefix of y_0. Thus, if $a \neq 1$ we may write $a = a_0 \circ a_1$ and $y_0 = a_1^{-1} \circ y_0'$, where $ay_0 = a_0 \circ y_0'$ and $a_0 \neq 1$. Since a is accepted by A we have $\tau(ay_0) = \tau(a_0y_0') = \tau(y_0)$. By definition of s_0 the first letter of e_0 is not readable from the vertex $\tau(y_0)$, so it follows, as before that $a_0y_0'e = a_0 \circ y_0' \circ e$. Similarly, if $b \neq 1$ then $b = b_0 \circ b_1$, where $b_0 \neq 1$, and $z_0 = b_0 \circ z_0'$, with $z_0^{-1}b = (z_0')^{-1} \circ b_1$ and $e(z_0')^{-1}b_1 = e \circ (z_0')^{-1} \circ b_1$. This implies that $s_1 = a_0 \circ y_0' \circ e \circ (z_0')^{-1} \circ b_1$ and by considering maximal L_Q-prefixes of both sides we see that $y_1 = a_0 \circ y_0'$ and $z_1 = b_1^{-1} \circ z_0'$. We now have $\tau(y_1) = \tau(a_0y_0') = \tau(y_0)$ which means that $y_0 = y_1$ and $a_0 = a_1^{-1}$, a contradiction. Hence $a = 1$. Similarly $z_0 = z_1$ gives rise to a contradiction, so $b = 1$. Hence $s_0 = s_1$ in this case.

Finally consider the case where s_1 and s_2 belong to $S^{(2)}$. As $s_1 = as_0b$, with $a, b \in H$, Lemma lem:equiv_verts 2.3 and the definition of $S^{(2)}$ imply that $s_0 = s_1$. Therefore S is a set of double coset representatives. □

ex:f_1 **Example 2.7.** Let F_1 be the free group on generators x_1, x_2, x_3 and H_1 its subgroup $H_1 = \langle h_1, h_2, h_3 \rangle$, where $h_1 = x_1^3$, $h_2 = x_2x_3x_2^{-1}$ and $h_3 = x_1x_2x_3$.

The Stallings automata, Γ_{A_1} for H_1, with maximal subtree T_1 highlighted and base vertex 1, is shown in Figure fig:stall1 2.1a. The set L_{T_1} corresponding to the maximal subtree T_1 is $L_{T_1} = \{1, x_1, x_1^{-1}, x_2, x_3^{-1}\}$.

Let the set of double coset representatives for H_1 be $S_1 = S_1^{(1)} \cup S_1^{(2)}$. To find the double coset normal form for the element $f_1 = x_2x_3^2x_2^{-1}x_1^4x_3^3x_1^{-2}x_3^{-1}x_2^{-1}x_1^{-1}$: in the notation of Algorithm I, use A_1 to read off the maximal acceptable prefix $h = x_2x_3^2x_2^{-1}x_1^3 = h_2^2h_1$ of f_1; and the maximal L_{Q_1} prefix $p = x_1$ of the remaining part of f_1. Next $q^{-1} = x_1x_2x_3x_1^2x_3^{-3}$, which has maximal acceptable prefix $g = x_1x_2x_3 = h_3$. The maximal L_{Q_1}-prefix of the remaining part of q^{-1} is then $t = x_1^2$. Setting $e = x_3$,

$$f_1 = h \circ p \circ e \circ t^{-1} \circ g^{-1}.$$

As p is an L_{T_1}-prefix of pq we have $y = p$. On the other hand t is not an L_{T_1}-prefix of q^{-1} and $\tau(t) = 2$, so $z = w(2) = x_1^{-1}$. Now set

$$s = p \circ e \circ z^{-1} = x_1x_3x_1 \in S_1^{(1)}.$$

Then, writing $h' = gtz^{-1} = h_3h_1$, the normal form for f_1 is

$$f_1 = hs(h')^{-1} = (h_2^2h_1)x_1x_3x_1(h_1^{-1}h_3^{-1}) \in H_1S_1^{(1)}H_1.$$

To find the double coset normal form for the element $f_2 = x_3^{-1}x_2^{-1}x_1^{-1}x_2x_3^{-1}x_2^{-1}x_1^{-1}$: use A_1 to read off the maximal acceptable prefix $h = x_3^{-1}x_2^{-1}x_1^{-1}x_2x_3^{-1}x_2^{-1} = h_3^{-1}h_2^{-1}$ of f_2; and the maximal L_{Q_1}-prefix $p = x_1^{-1}$ of the rest of f_2. Here $q^{-1} = 1$ and so $g = t = 1$, and

$$f_2 = h \circ p.$$

Therefore f_2 will be in $H_1S_1^{(2)}H_1$. To find the required double coset representative we construct $\Gamma_{A_1} \times \Gamma_{A_1}$, which has two non-diagonal components, shown in Figures 2.1b and 2.1c, with the $\sim$ representatives $(3,1)$ and $(2,1)$, shown as solid vertices. The connecting elements are labels of paths in the highlighted subtrees. All other non-diagonal components consist of isolated vertices. Now, $u = \tau(p) = 2$ and $v = \tau(t) = 1$ and $(2,1)$ is a $\sim$ representative. Hence the connecting element $c = c(2,1) = 1$. Therefore $p = y = w(2) = x_1^{-1}$, $z = t = w(1) = 1$, $yz^{-1} = x_1^{-1} \in S_1^{(2)}$,

$$a = hpcy^{-1} = h_3^{-1}h_2^{-1}x_1^{-1}x_1 = h_3^{-1}h_2^{-1} \text{ and } b = zc^{-1}t^{-1}g^{-1} = 1,$$

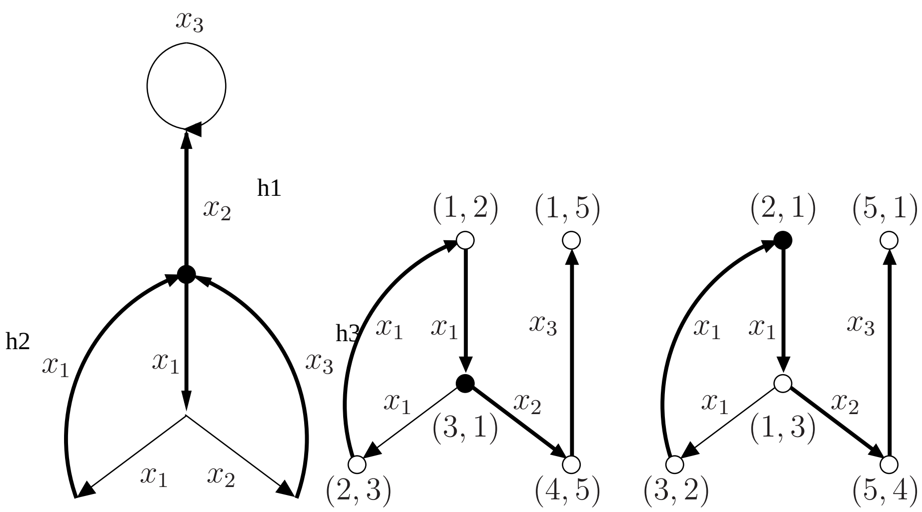
and the normal form of f_2 is

$$f_2 = ayz^{-1}b = (h_3^{-1}h_2^{-1})x_1^{-1}.$$

ex:f_2 **Example 2.8.** Let F_2 be the free group on generators y_1, y_2, y_3, y_4 with the subgroup $H_2 = \langle h_1', h_2', h_3' \rangle$, where $h_1' = y_2^2$, $h_2' = y_3y_4$ and $h_3' = y_1^2y_3y_1^{-1}y_2$. The Stallings automata, Γ_{A_2} for H_2, with maximal subtree T_2 highlighted and base vertex 1, is shown in Figure 2.2a. The set L_{T_2} corresponding to the maximal subtree T_2 is $L_{T_2} = \{1, y_1, y_1^2, y_2^{-1}, y_2^{-1}y_1, y_3\}$. Let the set of double coset representatives for H_2 be $S_2 = S_2^{(1)} \cup S_2^{(2)}$.

To find the normal form of $f_1' = y_1^2y_3y_1^{-1}y_2y_3y_4y_1y_2y_3y_4y_2^2$: use A_2 to read off the maximal acceptable prefix $h = y_1^2y_3y_1^{-1}y_2y_3y_4 = h_3'h_2'$ of f_1'; and the maximal L_{Q_2}-prefix $p = y_1$ of the rest of f_1'. Next $q^{-1} = y_2^{-2}y_4^{-1}y_3^{-1}y_2^{-1}$, so $g = y_2^{-2}y_4^{-1}y_3^{-1} = h_1'^{-1}h_2'^{-1}$ and $t = y_2^{-1}$. This element will be represented by an element of $S_2^{(2)}$, so we need to construct $\Gamma_{A_2} \times \Gamma_{A_2}$. There are 7 non-trivial, non-diagonal components. One is shown in Figure 2.2b and the remaining six in Figure 2.3. In all cases the solid vertex corresponds to the $\sim$ representative and connecting elements are paths in the highlighted trees. Following Algorithm I, the normal form of f_1' is

$$f_1' = hyz^{-1}g^{-1},$$



(a) Stallings automaton Γ_{A_1} for H_1 fig:stall1

(b) $\Gamma_{A_1} \times \Gamma_{A_1}$: connected component of $(3, 1)$ fig:GxG-1

(c) $\Gamma_{A_1} \times \Gamma_{A_1}$: connected component of $(2, 1)$ fig:GxG-2

fig:stall Figure 2.1: Example ex:f_1 2.7.

where $y = y_1$, $z = y_2$ and $yz^{-1} = y_1 y_2^{-1} \in S_2^{(2)}$: that is

$$f'_1 = h'_3 h'_2 y_1 y_2^{-1} (h'_1)^{-1} (h'_2)^{-1}.$$

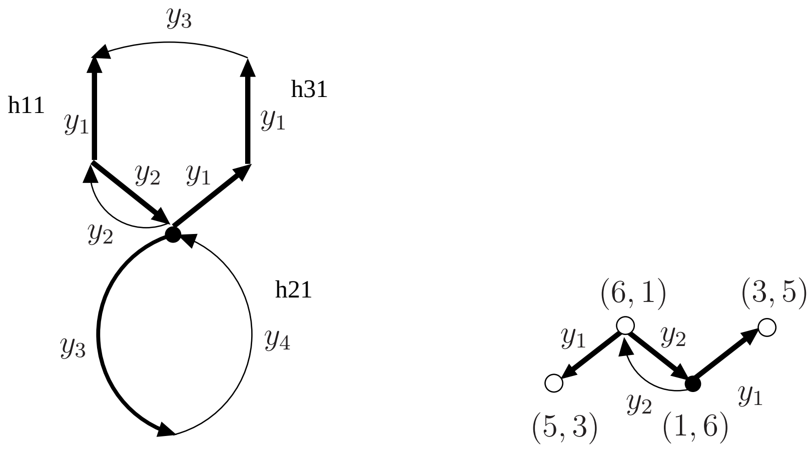
In fact if we don't require uniqueness of representatives then there is a much simpler algorithm to write words in normal form. This simply finds the maximal prefix h_1 of w accepted by A and so $w = h_1 \circ v$. It then finds the maximal prefix p of v^{-1} accepted by A. Setting $h_2 = p^{-1}$ gives $w = h_1 \circ d \circ h_2$, for some uniquely determined word d, which is a (non-unique) double coset representative.

3 The generalised folding process

sec:foldings

Recall from above that F_1 and F_2 are free groups on finite sets X_1 and X_2, respectively; $H_1 \leq F_1$ and $H_2 \leq F_2$ are subgroups of rank m, freely generated by $\{h_1, \ldots, h_m\}$ and $\{h'_1, \ldots, h'_m\}$. The isomorphism from H_1 to H_2 which maps h_i to h'_i is denoted ϕ.

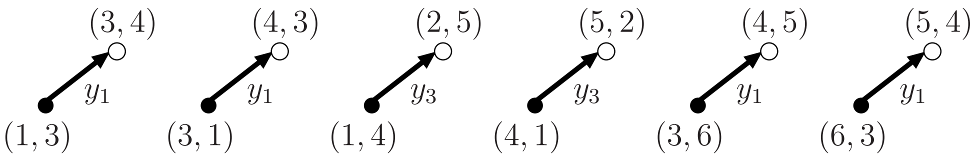
Now let $Z = \{z_1, \ldots, z_m\}$ be a set disjoint from $X_1 \cup X_2$. Then there is an isomorphism $\phi_1 : \mathbb{F}(Z) \to H_1$, such that $\phi_1(z_i) = h_i$, and an isomorphism $\phi_2 : \mathbb{F}(Z) \to H_2$, such that $\phi_2(z_i) = h'_i$, $i = 1, \ldots, m$. Let G to be the free



(a) Stallings automaton A_2 for H_2 fig:stallh2

(b) $\Gamma_{A_2} \times \Gamma_{A_2}$: connected component of $(6,1)$ fig:G2xG2-1

Figure 2.2: Stallings automata for Example 2.8. ex:f_2 fig:stallagain



(a) $(1,3)$ fig:G2xG2-2-1 (b) $(3,1)$ fig:G2xG2-2-2 (c) $(2,5)$ fig:G2xG2-2-3 (d) $(5,2)$ fig:G2xG2-2-4 (e) $(4,5)$ fig:G2xG2-2-5 (f) $(6,3)$ fig:G2xG2-2-6

Figure 2.3: Example 2.8: connected components of $\Gamma_{A_2} \times \Gamma_{A_2}$. ex:f_2 fig:G2xG2-2

product with amalgamation: $G = F_1 \underset{H_1=H_2}{*} F_2$. Then G has a presentation

$$\langle X_1 \cup X_2 \cup Z | h_i = z_i = h'_i, i = 1 \dots m \rangle.$$

def:dcnf **Definition 3.1.** *Let S_1 and S_2 be sets of double coset representatives for $H_1 \leq F_1$ and $H_2 \leq F_2$, respectively. A word $w \in \mathbb{F}(X_1 \cup X_2 \cup Z)$ is in* double coset normal form *(or* dc-normal form*) if $w = h_0 p_1 h_1 p_2 \cdots h_{k-1} p_k h_k$, $k \geq 0$, where for $i = 1, \dots, k$,*

(a) h_i is a reduced word in $\mathbb{F}(Z)$, for $i = 0, \dots, k$;

(b) $p_i \in S_1 \cup S_2$ and $p_i \neq 1$, for $i = 1, \dots, k$, and

(c) if $p_i \in S_j$ then $p_{i+1} \notin S_j$, $j \in \{1, 2\}$.

From now on when we say "normal form" we mean "double coset normal form" unless we explicitly say something to the contrary.

thm:dcnf **Theorem 3.2.** *Every element of G is represented by a unique element of $\mathbb{F}(X_1 \cup X_2 \cup Z)$ in double coset normal form.*

Proof. To see that every $g \in G$ can be written in dc-normal form, first write g in reduced form; say $g = f_1 \cdots f_t$, where f_i is in a factor. Assuming that $f_i \in F_j$, let $a_i s_i b_i$ be the double coset representative of f_i, so $a_i, b_i \in H_j$ and $s_i \in S_j$. Let $z'_i = \phi_j^{-1}(a_i)$ and $z''_i = \phi_j^{-1}(b_i)$. Repeat this for $i = 1, \dots, t$ and then let z_i be the free reduction of $z'_i z''_{i+1}$, for $i = 1, \dots, t-1$. Setting $z_0 = z'_1$ and $z_t = z''_t$, it follows that

$$z_0 s_1 z_1 \cdots z_{t-1} s_t z_t$$

is a dc-normal form for g.

Now suppose that w and w' are words in dc-normal form and that $w =_G w'$. Let $w = h_0 p_1 h_1 p_2 \cdots h_{k-1} p_k h_k$, and $w' = h'_0 p'_1 h'_1 p'_2 \cdots h'_{k-1} p'_k h'_{k'}$, where $h_i, h'_i \in \mathbb{F}(Z)$ and $p_i, p'_i \in S_1 \cup S_2$. For fixed $i \geq 2$, assuming that $p_i \in F_j$, let $f_i = p_i \phi_j(h_i)$. Similarly let $f_1 = \phi_j(h_0) p_1 \phi_j(h_1)$. Then $f_1 \cdots f_k$, is a reduced form for the element $w_1 \in G$. Similarly, we obtain a reduced form $f'_1 \cdots f'_{k'}$ for w'. If $k = 0$ then we may assume $w =_G f_1 = \phi_1(h_0) \in H_j$, and it follows that $k' = 0$ and $w' =_G f'_1 = \phi_1(h'_1) \in H_1$. As H_1 embeds in G this implies that $w = w'$. Thus the result holds when $k = 0$ and we may assume inductively that $k > 0$ and the result holds for elements represented by normal forms of length less than k. Comparing reduced forms, we have $k = k'$.

Moreover, $1 =_G = w' w^{-1} = f'_1 \cdots f'_k f_k^{-1} \cdots f_1^{-1}$, so by the fundamental theorem for reduced forms, we have $f'_k f_k^{-1} \in H_j$, for $j = 1$, or 2. Hence, for some $a, b \in H_j$ we have $p'_k = a p_k b$. Therefore $p'_k = p_k$, and now, as p_k is a double coset representative and $f'_k f_k^{-1} \in H_j$, it follows that $h_k = h'_k$. Applying the inductive hypothesis to their prefixes of length $2k-1$, we see that $w = w'$, as required. □

The object of this section is to construct an automaton which will accept a word w in double coset normal form if and only if it belongs to a given subgroup K of G. The idea is to do this by starting with the flower automaton for the generators of K, written in (double coset) normal form; carrying out Stallings folding as usual to produce an inverse automaton; and next adding some additional paths to allow normal forms to be read. This may introduce new non-determinism in some states (i.e. edges which may be folded), so the resulting automaton must be folded again. The result of the final folding is the candidate automaton.

Suppose L is the language accepted by the flower automaton of K. The image of L under the canonical map to G is K. All the stages of our generalised folding process will preserve this image, so our final automaton will accept a language L_1 which also maps to K. We must then prove that, if w is a word in normal form which represents an element of K then w is in L_1. It will therefore suffice to show that, if u is any word in L_1 then the normal form of u is also in L_1.

Let S_1 and S_2 be sets of double coset representatives of $H_1 \le F_1$ and $H_2 \le F_2$, respectively, as constructed in Section sec:dcforms 2. Denote $S = S_1 \cup S_2$. Let A_k be the Stallings automaton for H_k and let T_k be a spanning tree for the associated graph Γ_{A_k}, $k = 1, 2$. The alphabet of A_k is X_k and the set of states of A_k is denoted Q_k.

Suppose that $g \in G$ and $g = g_1 \cdots g_t$ is in reduced form. Write each syllable g_i of $g = g_1 \cdots g_t$ in normal form using the algorithm I above. This gives $g_i = h_{i,1} d_i h_{i,2}$, with $d_i \in S$ and $h_{i,j} \in H_1 \cup H_2$. Using ϕ_1^{-1} or ϕ_2^{-1}, as appropriate, we now write $h_{i,1}$ and $h_{i,2}$ as reduced words in $\mathbb{F}(Z)$. For $i = 1, \ldots, t-1$, we reduce the word $h_{i,2} h_{(i+1),1} \in \mathbb{F}(Z)$ to give a reduced word $h_i \in \mathbb{F}(Z)$ and set $h_0 = h_{1,1}$ and $h_{t+1} = h_{t,2}$. Then g has normal form $h_0 d_1 h_1 \cdots d_t h_{t+1}$.

ex:g **Example 3.3.** Let F_i and H_i be given in Examples ex:f_1 2.7 and Example ex:f_2 2.8, and let f_1 and f_2 be the elements of F_1 in Example ex:f_1 2.7 and $f'_1 \in F_2$ as in Example ex:f_2 2.8. Set $z_i = h_i = h'_i$ for $i = 1, 2, 3$.

Let $g = f_1 f_1' f_2$; then we have

$$\begin{aligned} f_1 &= (h_2^2 h_1) x_1 x_3 x_1 (h_1^{-1} h_3^{-1}) \\ &= z_2^2 z_1 x_1 x_3 x_1 z_1^{-1} z_3^{-1}, \\ f_2 &= (h_3^{-1} h_2^{-1}) x_1^{-1} \\ &= z_3^{-1} z_2 x_1^{-1} \text{ and} \\ f_1' &= h_3' h_2' y_1 y_2^{-1} (h_1')^{-1} (h_2')^{-1} \\ &= z_3 z_2 y_1 y_2^{-1} z_1^{-1} z_2^{-1}. \end{aligned}$$

Therefore the double coset normal form of g is

$$g = z_2^2 z_1 x_1 x_3 x_1 z_1^{-1} z_2 y_1 y_2^{-1} z_1^{-1} z_2^{-1} z_3^{-1} z_2^{-1} x_1^{-1}.$$

3.1 Double coset automata

sec:dca

Let $K = \langle k_1, \ldots, k_s \rangle$, where k_i is an element of G written in normal form: say

$$k_i = h_{i,0} t_{i,1} h_{i,1} \cdots t_{i,m_i} h_{i,m_i+1}, \tag{3.1}$$

eq:k-form

with $h_{i,j} \in \mathbb{F}(Z)$ and $t_{i,j} \in S_1 \cup S_2$. Denote by $\hat{K}$ the subgroup of $\mathbb{F}(X_1 \cup X_2 \cup Z)$ generated by $k_1, \ldots, k_s$. Let $\Sigma = (X_1 \cup X_2 \cup Z)^{\pm 1}$. Let $\mathcal{F}(K)$ be the flower automaton of $\hat{K}$, and let Γ be the corresponding rooted graph. If e is an edge of Γ then e is labelled by a letter of Σ occuring in $h_{i,j}$ or $t_{i,j}$, for some i, j, as in (eq:k-form 3.1). If e is labelled by a letter of Z we say e has *type* Z. If e is labelled by a letter of X_k occuring in $t_{i,j}$, we say e has *type* X_k, $k = 1, 2$.

Now let Γ_K be the Stallings folding of the graph Γ. Then Γ_K is inverse and $\pi(L(\Gamma_K)) = K$. However $L(\Gamma_K)$ does not, in general, contain all normal forms of elements of K. We wish to transform Γ_K to allow it to accept normal forms of elements of K. In outline this transformation process consists of running the loop below, on input Γ_K, until it halts, at which point we shall show that we have an inverse automaton which accepts normal forms of elements of K. Assume then that $\Delta_{(0)}$ is an inverse automaton, with alphabet Σ and $\pi(L(\Delta_{(0)})) = K$.

Loop. Set $n = 0$.

it:gf1 Step 1. Apply Algorithm II below to add new paths to $\Delta_{(n)}$ which allow normal forms of labels of certain paths to be read. Call the result $\Delta''_{(n)}$.

it:gf2 Step 2. Construct the Stallings automaton $\Delta_{(n+1)}$ of $\Delta''_{(n)}$. If $\Delta_{(n+1)}$ and $\Delta_{(n)}$ have the same number of X_1 and X_2 components (see below) then output $\Delta_{(n+1)}$ and stop. Otherwise, add 1 to n and repeat Step it:gf1 Step 1..

Algorithm II below describes exactly how to carry out Step 1 of this loop. Since Step 1 may run more than once, the input to this algorithm is assumed to be an arbitrary inverse automaton Δ, with alphabet Σ and $\pi(L(\Delta)) = K$. The output of Algorithm II is a rooted, labelled, involutive automaton, with the same alphabet, which may fail to be deterministic. We shall show that, if we start as above with $\Gamma_K = \Delta_{(0)}$, the loop halts, at some $n < \infty$; at which point $\Delta_{(n+1)}$ is an inverse automaton which accepts the normal form of every element of K.

Algorithm II.

Let Δ be an inverse automaton, with alphabet Σ and start and final state 1, such that $\pi(L(\Delta)) = K$. For $k = 1, 2$, let Δ_k be the graph formed from Δ by removing all edges of type $X_{k'}$, where $k \neq k'$. An X_k component of Δ_k is the subgraph of Δ formed from a connected component of Δ_k by removing all leaves which are incident to edges of type Z, and then repeating the process till there are no such leaves left. Given an X_k component Θ of Δ_k, a *boundary vertex* of Θ is defined to be a vertex α such that α is incident, in Δ, to a vertex which does not belong to Θ. We shall modify each X_k component Θ of Δ_k so that if p is a path, from a boundary vertex u to a boundary vertex v of Θ, then the normal form of the label $l(p)$ of p is the label of a path from u to v. Throughout the modification process we keep track of the boundary vertices, so that once modification is complete the X_k components can be reassembled, by attaching as they did in Δ. The modification of an X_k component Θ of Δ_k consists of five steps, producing new graphs Θ_1, Θ_2, Θ_3, Θ_4 and Θ_5. In each case there is a canonical morphism θ_i from Θ_{i-1} to Θ_i and, (writing $\Theta = \Theta_0$) we define the *boundary vertices* of Θ_i to be the vertices $\theta_i(v)$ of Θ_i, such that v is a boundary vertex of Θ_{i-1}. Moreover the images $\theta_i \circ \cdots \circ \theta_1(v)$ of vertices of Θ in Θ_i will be referred to as *vertices of* Θ.

The modification process is followed by a "reassembly" phase, in which we reconnect the X_k components of Δ, to form the output of the algorithm.

Modification 1: $\Theta \rightsquigarrow \Theta_1$.
This step involves the addition of paths labeled by X_k-words corresponding to z-edges of an X_k component.

Let Θ be an X_k component of Δ_k. If (α, z, β) is an edge of Θ labelled by an element $z \in Z$ then add a path from α to β labelled by $\phi_k(z)$ to the graph Θ. Do this for all such edges and call the result Θ_1'. Fold fig:alg2-1a fig:alg2-1b

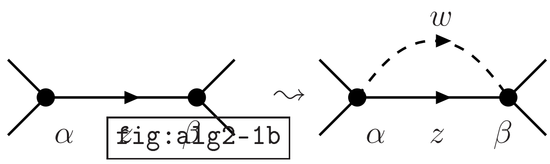


fig:alg2-1

Figure 3.1: Θ to Θ_1: $w = \phi_k(z)$.

Θ_1' to form the graph Θ_1. (See Figure fig:alg2-1 3.1.) Let θ_1 be the canonical morphism from Θ to Θ_1. If α and β are vertices of Θ then, since θ is a morphism $\pi(L(\Theta,\alpha,\beta)) \subseteq \pi(L(\Theta_1,\theta(\alpha),\theta(\beta)))$. On the other hand if w is in $L(\Theta_1,\theta(\alpha),\theta(\beta))$ then, since Θ_1 is a folding of Θ_1', there is a path labelled w' from α to β in Θ_1', such that $\pi(w') = \pi(w)$. By construction then there exists a path w'' from α to β in Θ, such that $\pi(w'') = \pi(w')$. Hence $\pi(L(\Theta_1,\theta(\alpha),\theta(\beta))) = \pi(L(\Theta,\alpha,\beta))$.

Modification 2: $\Theta_1 \rightsquigarrow \Theta_2$**.**
Here we add to Θ_1 paths labelled by ϕ_k^{-1} images of labels of simple paths in $\Theta_1 \times \Gamma_{A_k}$, from $(\alpha_i, 1)$ to $(\alpha_j, 1)$; for appropriate i, j. Let $\mathcal{P}_1 = \Theta_1 \times \Gamma_{A_k}$. Then there is a path labelled w from α to β in Θ_1 and a path labelled w from α' to β' in Γ_{A_k} if and only if there is a path labelled w from (α,α') to (β,β') in $\mathcal{P}_1$. We shall use this property of $\mathcal{P}_1$ to determine which new paths to add to Θ_1. Choose a spanning forest Υ_1 of $\mathcal{P}_1$. Next order the vertices of Θ_1: say these are $\alpha_0, \dots, \alpha_t$, in the order written.

Let α_i, α_j be vertices of Θ with $i \le j$. If

- there exists a simple path p in $\mathcal{P}_1$ from $(\alpha_i, 1)$ to $(\alpha_j, 1)$ and
- p does not contain a vertex $(\alpha_k, 1)$ with $k \in \{i, j\}$

then let $l_p \in \mathbb{F}(X_k)$ be the label of p and let $w_p = \phi_k^{-1}(l_p) \in \mathbb{F}(Z)$. Let q be a path (disjoint from Θ_1) of length $|w_p|$ and with label w_p. Identify the initial vertex of q with α_i and the terminal vertex of q with α_j. (See Figure fig:alg2-2 3.2.) Repeat this process for all simple paths from $(\alpha_i, 1)$ to $(\alpha_j, 1)$, over all pairs of vertices α_i, α_j of Θ, with $i \le j$. There are finitely many simple paths in $\mathcal{P}_1$ so this process terminates. Call the result Θ_2'. Then Θ_1 is a subgraph of Θ_2'. Now fold Θ_2' to give a new graph Θ_2. The composition θ_2 of the the embedding map of Θ_1 into Θ_2' with the canonical morphism from Θ_2' to Θ_2 is a morphism from Θ_1 to Θ_2. Moreover, if α and β are vertices of Θ_1 and a path q from α to β, with label w_p, is added to Θ_1 in forming Θ_2, then there is a path p in Θ_1 with $\pi(l(p)) = \pi(w_p)$. Thus the argument of the previous case shows that $\pi(L(\Theta_1,\alpha,\beta)) = \pi(L(\Theta_2,\theta_2(\alpha),\theta_2(\beta)))$.

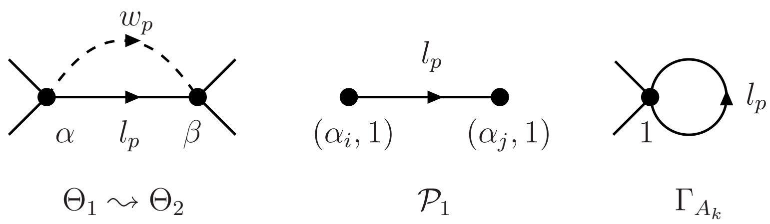


fig:alg2-2

Figure 3.2: Θ_1 to Θ_2: $w_p = \phi_k^{-1}(l_p)$.

Modification 3: $\Theta_2 \rightsquigarrow \Theta_3$.
At this stage we add ϕ_k^{-1} images of paths in $\Theta_2 \times \Gamma_{A_k}$ related to edges of this graph which do not belong to its spanning subforest and which project to closed paths, based at 1, in Γ_{A_k}.

As the edges added to Θ_1 to form Θ_2 are all labelled by elements of Z, and all edges of Γ_{A_k} are labelled by elements of X_k the graphs $\mathcal{P}_1 = \Theta_1 \times \Gamma_{A_k}$ and $\mathcal{P}_2 = \Theta_2 \times \Gamma_{A_k}$ differ only in that the second may have some new isolated vertices. Therefore we may choose a spanning forest Υ_2 of $\mathcal{P}_2$ consisting of the spanning forest Υ_1 of $\mathcal{P}_1$ with some new isolated vertices if necessary. If $\mathcal{P}_2$ is a forest then $\mathcal{P}_2 = \Upsilon_2$, there is nothing to do at the this stage, and we immediately set $\Theta_3 = \Theta_2$. Otherwise, suppose that e is an edge of $\mathcal{P}_2$ which does not belong to Υ_2 but which does belong to a component Ξ of $\mathcal{P}_2$ containing a vertex $(\alpha, 1)$, for some vertex α of Θ. Let $(\alpha, 1)$ be such a vertex and let $e = ((\alpha_i, \beta_1), x, (\alpha_j, \beta_2))$ be an edge in the same component of $\mathcal{P}_2$ as $(\alpha, 1)$, with label x in X_k. Let p_i be the path in Υ_2 from $(\alpha, 1)$ to (α_i, β_1) and let p_j be the path in Υ_2 from (α_j, β_2) to $(\alpha, 1)$ and let l_i and l_j be the labels of p_i and p_j, respectively. Then p_i, e, p_j projects to a closed path in Γ_{A_k}, based at 1, with label $h_e = l_i x l_j \in H_k \subseteq F_k$. Moreover p_i, e, p_j projects to a closed path in Θ_2, based at α, also with label h_e. Let $w_e = \phi_k^{-1}(h_e) \in \mathbb{F}(Z)$ and let q be a path (disjoint from Θ_2) of length $|w_e|$ and with label w_e. Identify the initial and terminal vertices of q to the vertex α of Θ_2. (See Figure fig:alg2-3 3.3.) Repeat this process for all such edges e and vertices $(\alpha, 1)$ of $\mathcal{P}_2$, fold the resulting graph, and denote the result by Θ_3. (In practice it's not necessary to repeat this process for *all* such vertices $(\alpha, 1)$ in Ξ: but it makes some arguments later easier if we assume that we do so.) As in the previous case there is a natural morphism θ_3 from Θ_2 to Θ_3. Again, if q is a path added to Θ_2 in forming Θ_3 then there is a path p in Θ_2, with the same end points as q, such that $\pi(l(p)) = \pi(l(q))$. Hence, as before, for all vertices α, β of Θ_2, $\pi(L(\Theta_2, \alpha, \beta)) = \pi(L(\Theta_3, \theta_3(\alpha), \theta_3(\beta)))$.

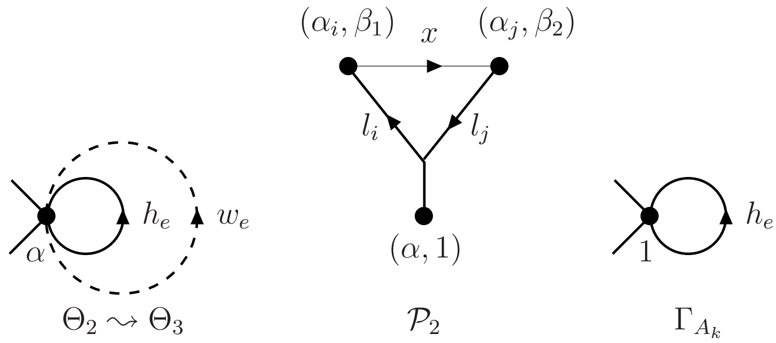


fig:alg2-3

Figure 3.3: Θ_2 to Θ_3: $h_e = l_i x l_j$, $w_e = \phi_k^{-1}(h_e)$.

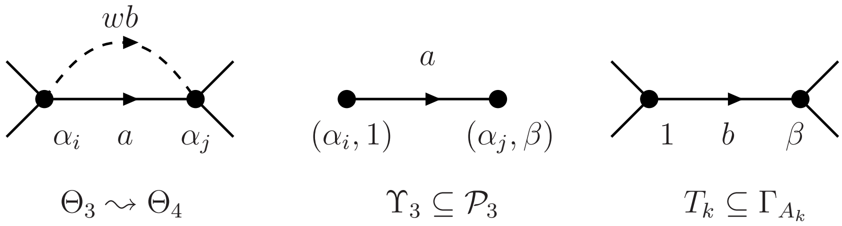


fig:alg2-4

Figure 3.4: Θ_3 to Θ_4: $w = \phi_k^{-1}(ab^{-1})$.

Modification 4: $\Theta_3 \rightsquigarrow \Theta_4$.
Next we wish to add paths to Θ_3 that allow us to read the normal forms of words w which are readable by Θ_3 and readable, but not accepted, by Γ_{A_k}. Let $\mathcal{P}_3 = \Theta_3 \times \Gamma_{A_k}$. As before we may choose a spanning forest Υ_3 of $\mathcal{P}_3$ which consists of Υ_2 and some additional isolated vertices.

Recall that we have fixed a spanning subtree T_k of Γ_{A_k}. Let $\delta = (\alpha_j, \beta)$ be a vertex of $\mathcal{P}_3$, with $\beta \neq 1$, which lies in a connected component of $\mathcal{P}_3$ containing a vertex $\gamma = (\alpha_i, 1)$, for some vertices α_i and α_j of Θ. Let b be the label of the path in T_k from 1 to β. If $\mathcal{P}_3$ contains a simple path from $(\alpha_i, 1)$ to (α_j, β), with label b, then say that $\mathcal{P}_3$ *covers* the pair γ, δ. If all such pairs of vertices are covered by $\mathcal{P}_3$ then set $\Theta_4 = \Theta_3$. Otherwise let p be the simple path in Υ_3 from a vertex $\gamma = (\alpha_i, 1)$ to a vertex $\delta = (\alpha_j, \beta)$, where γ, δ is not covered by $\mathcal{P}_3$. Let p have label a. Then $ab^{-1} = h \in H_k$. Let $w = \phi_k^{-1}(ab^{-1}) \in \mathbb{F}(Z)$ and let q be a path (disjoint from Θ_3) with label wb. Identify the initial and terminal vertices of q with vertices α_i and α_j of Θ_3, respectively, to form a new graph Θ_3'. (See Figure fig:alg2-4 3.4.)

As $\pi(a) = \pi(wb) \in G$, if L_3 and L_3' are the languages accepted by $(\Theta_3, \alpha, \beta)$ and $(\Theta_3', \alpha, \beta)$, for some vertices α, β of Θ_3, then $\pi(L_3) = \pi(L_3')$. Repeat this process for all pairs of vertices which are not covered by $\mathcal{P}_3$ and fold the result to form the graph Θ_4. Again there is a natural mor-

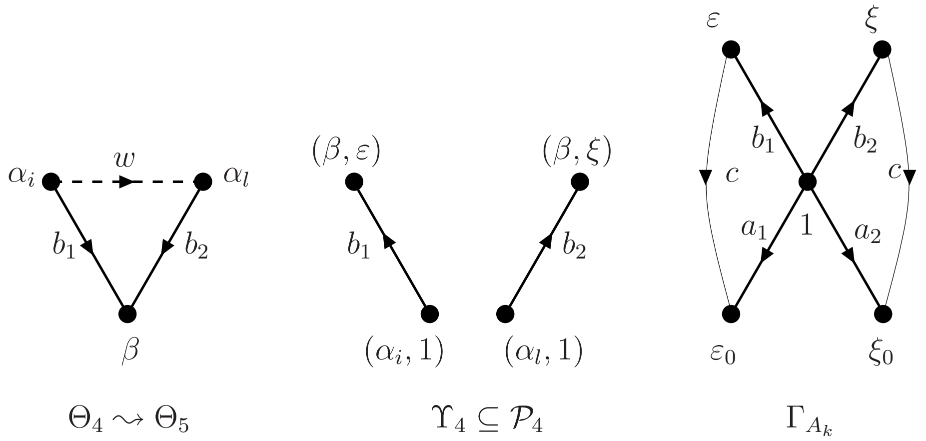


fig:alg2-5

Figure 3.5: Θ_4 to Θ_5: $w = \phi_k^{-1}(b_1 c a_1^{-1}) a_1 a_2^{-1} \phi_k^{-1}(a_2 c^{-1} b_2^{-1})$.

phism θ_4 from Θ_3 to Θ_4. If L_4 is the language accepted by $(\Theta_4, \theta_4(\alpha), (\beta))$, for some vertices α, β of Θ_3, then we have $\pi(L_3) = \pi(L_4)$, as in previous cases.

Modification 5: $\Theta_4 \rightsquigarrow \Theta_5$**.**
Finally we wish to add paths which allow double coset representatives of type 2 to be read. Let $\mathcal{P}_4 = \Theta_4 \times \Gamma_{A_k}$ and choose a spanning forest Υ_4 for $\mathcal{P}_4$.

Suppose $(\varepsilon, \xi) \in P \subseteq V(\Gamma_{A_k} \times \Gamma_{A_k})$ and that the $\sim$ representative of (ε, ξ) is (ε_0, ξ_0). Let $b_1 = w(\varepsilon)$, $b_2 = w(\xi)$ (the labels of paths from 1 to ε and 1 to ξ in the subtree T_k of Γ_{A_k}). Let $a_1 = w(\varepsilon_0)$ and $a_2 = w(\xi_0)$. By definition of $\sim$ there are paths in Γ_{A_k}, with label $c = c(\varepsilon, \xi)$, from ε to ε_0 and from ξ to ξ_0. Furthermore $h_i = b_i c a_i^{-1} \in H_k$ and $w_i = \phi_k^{-1}(h_i) \in \mathbb{F}(Z)$, for $i = 1, 2$. If there exist paths p_1, with label b_1 from $(\alpha_i, 1)$ to (β, ε), and p_2, with label b_2 from $(\alpha_l, 1)$ to (β, ξ), in Υ_4, then let q be a path (disjoint from Θ_4) with label $w_1 a_1 a_2^{-1} w_2^{-1}$, and identify the initial and terminal vertices of q with vertices α_i and α_l of Θ_4, respectively. (See Figure 3.5.)

Repeat this process for all such paths p_i and p_j and all such pairs (ε, ξ). To see that the image of the language accepted by the new graph is the same as that of the original note that the word $b_1 b_2^{-1}$ is readable, starting at α_i and ending at α_l, in Θ_4. As $\pi(b_1 b_2^{-1}) = \pi(w_1 a_1 a_2^{-1} w_2^{-1})$ the addition of this new path has no effect on the image, under π, of the language accepted by the automaton. Fold the resulting graph to form Θ_5. As before, there is a natural morphism θ_5 from Θ_4 to Θ_5 and again, if α, β are vertices of Θ_4 and L_4 and L_5 are the languages accepted by $(\Theta_4, \alpha, \beta)$ and $(\Theta_5, \theta_4(\alpha), \theta_4(\beta))$, respectively, then $\pi(L_4) = \pi(L_5)$.

Reassembly.
The modification process is applied to all X_k components of Δ_k, for $k = 1$ and 2. Roughly speaking a new graph is then constructed by reconnecting the modified X_k components in the same way as they were connected in Δ. In detail, let $\mathcal{C}$ be the set of all X_1 and X_2 components. Define

$$E_Z = E(\Delta) \setminus \cup_{\Phi \in \mathcal{C}} E(\Phi)$$

and Δ_Z to be the subgraph of Δ consisting of all edges of E_Z, and their incident vertices. Define Δ' to be the disjoint union of the graphs Θ, such that $\Theta \in \mathcal{C}$, with the graph Δ_Z. Also define Δ'_5 to be the disjoint union of the graphs Θ_5, such that $\Theta \in \mathcal{C}$, with the graph Δ_Z. For each $\Theta \in \mathcal{C}$ there is a morphism $\theta_5 \circ \theta_4 \circ \theta_3 \circ \theta_2 \circ \theta_1$ from Θ to Θ_5. Define θ to be the morphism from Δ' to Δ'_5 which consists of the union of these morphisms with the identity morphism from Δ_Z to Δ_Z. By construction, for each connected component Ξ of Δ' there is an embedding of Ξ into Δ. Let ν be the union of these embeddings over all components of Δ'.

The output Δ'' of Algorithm II is the quotient of Δ'_5 defined as follows. Let u and v be vertices of Δ'_5.

- If $\nu(\theta^{-1}(u)) \cap \nu(\theta^{-1}(v)) \neq \emptyset$ then identify vertices u and v. For $u \in V(\Delta'_5)$ let $[u]$ denote equivalence class of u under the equivalence relation on $V(\Delta'_5)$ generated by this identification.
- Δ'' has edge set

$$\{([u], a, [v]) : (u, a, v) \in E(\Delta'_5)\}.$$

- The root vertex of Δ'' is the image, in Δ'', of $\nu^{-1}(1)$, where 1 is the root of Δ.

lem:resol-quot

Lemma 3.4. *Let $[\cdot]$ be the equivalence relation on $V(\Delta'_5)$ described above. There is a morphism ρ from Δ to Δ'' given by*

- $\rho(u) = [\theta(\nu^{-1}(u))]$, *for $u \in V(\Delta)$, and*
- $\rho((u, a, v)) = (\rho(u), a, \rho(v))$, *for $(u, a, v) \in E(\Delta)$.*

Proof. First ρ must be shown to be well defined. That is, we must verify that $[\theta(\nu^{-1}(u))]$ is a vertex of Δ'' and that $[\rho(u), a, \rho(v)]$ is an edge. If $u \in V(\Delta)$ then either $u \in V(\Delta_Z)$ or u belongs to some X_k component of Δ. Hence $\nu^{-1}(u) \neq \emptyset$. If $u_1, u_2 \in \nu^{-1}(u)$, then, by definition of Δ'', $\theta(u_1)$ and $\theta(u_2)$

are equivalent. Thus $[\theta(\nu^{-1}(u)]$ is a vertex of Δ'', as required. Suppose that (u, a, v) is an edge of Δ and let $u_1 \in \nu^{-1}(u)$ and $v_1 \in \nu^{-1}(v)$. Then $\rho(u) = [\theta(u_1)]$ and $\rho(v) = [\theta(v_1)]$, so $[\rho(u), a, \rho(v)]$ is an edge of Δ''. Hence ρ is well-defined, and is, moreover, a graph morphism. □

Definition 3.5. *Let Δ be an inverse automaton. The result Δ'' of applying* it:gf1 *Step 1. above to Δ is called a* double coset resolution *or* dc-resolution *of Δ. The graph Ψ obtained by applying* it:gf2 *Step 2. to Δ'' is called a* double coset folding *or* dc-folding *of Δ. The composition $\hat{\rho}$ of the morphism $\rho : \Delta \to \Delta''$ with the folding morphism $\Delta'' \to \Psi$ is called the* dc-folding morphism.

4 Summary of the generalised folding algorithm

In this section we summarise the algorithms which we have described above, in a form suitable for the analysis of their complexity, in Section sec:TC 6 below. That is we summarise Algorithms I and II and the Loop of Section sec:dca 3.1. The Loop, which calls Algorithm II during its execution, is what we refer to as the "Generalised Folding Algorithm".

4.1 Summary of algorithm I

sub:sum_algI

4.1.1 Preproccessing.

Given $F_1 *_{H_1=H_2} F_2$ the steps in this subsection are carried out once for each group H_i. Once these steps have been completed then Algorithm I can be run, as many times as necessary, to find double coset normal forms.
Input.

- generators X of a free group $F(X)$.
- A finite set Y of elements of $F(X)$.

Output.

- Γ_A, the Stallings automaton for the subgroup $H = \langle Y \rangle$ of $F(X)$.
- L_T, the set of words corresponding to a maximal subtree T of Γ_A.
- The set P of non-diagonal elements of $V(\Gamma_A \times \Gamma_A)$ partitioned into equivalence classes of $\sim$ (i.e. vertices in the same connected components of $\Gamma_A \times \Gamma_A$).

- P_0 the set of representatives of equivalence classes of elements of P.
- C the set of connecting elements, one for each element of P.

Process. An upper bound for each step is given in brackets. Let $N = \sum y \in Y|y|$.

A1 Construct Γ_A.

A2 Construct a spanning tree T for Γ_A and simultaneously compute L_T.

A3 Construct $\Gamma_A \times \Gamma_A$.

A4 Find connected components of $\Gamma_A \times \Gamma_A$. This can be done by doing a BFS or DFS of $\Gamma_A \times \Gamma_A$, which constructs a spanning forest F at the same time as finding the set C of connecting elements, which can be taken to be paths in the spanning forest F.

A5 Construct the set P_0. This can be done easily using $L_T(A)$ and the forest F.

4.1.2 Execution of Algorithm I

Input.

- w, a reduced word in $F(X)$.

Output.

- Double coset normal form of w.

Process.

B1 Use Γ_A to find the maximal prefix h of w accepted. Let $w = h \circ f$.

B2 Use Γ_A to find the maximal L_Q-prefix p of f. Let $f = p \circ q$.

B3 Use Γ_A to find the maximal prefix g of q^{-1} accepted. Let $q = r^{-1} \circ g^{-1}$.

B4 Use Γ_A to find the maximal L_Q-prefix t of r. Let $r = t \circ e^{-1}$.

B5 If $e = 1$ go to step it:ss B9.

B6 Using L_T, set $y = w(\tau(p))$ and $z = w(\tau(t))$.

B7 Freely reduce hpy^{-1}, yez^{-1} and $zt^{-1}g^{-1}$ and call the results a, b and c.

B8 Output a, b, c and stop.

it:ss B9 (This step is reached only if $e = 1$.) Using P_0 and F find the representative (u_0, v_0) of $(\tau(p), \tau(t))$.

B10 Look up the connecting element $c(u, v)$.

B11 Let $y = w(u_0)$ and $z = w(v_0)$.

B12 Freely reduce $hpcy^{-1}$ and $zc^{-1}t^{-1}g^{-1}$ and call the results a and b.

B13 Output a, yz^{-1}, b.

4.2 Summary of Algorithm II

subs_summaryII

Here we assume $F_1 *_{H_1=H_2} F_2$ where F_k is generated by X_k, H_k is generated by $Y_k = \{h_{k,1}, \ldots, h_{k,m}\}$ and the Stallings automata Γ_{A_k} for H_k have been constructed. Let $\Sigma = (X_1 \cup X_2 \cup Z)^{\pm 1}$ and, for $k = 1, 2$, assume that we have constructed, at the preprocessing stage, the following.

1. A spanning tree T_k for Γ_{A_k}, the set $L_{T_k}$of words corresponding to the maximal subtree T_k, the set P_k of non-diagonal elements of $V(\Gamma_{A_k} \times \Gamma_{A_k})$, the set of representatives $P_{k,0}$ of equivalence classes of elements of P_k and the set C_k of connecting elements, one for each element of P_k.

2. $\phi_k : Z \to F_k$ be a map inducing an isomorphism from $F(Z)$ to H_k. This map is encoded as part of Γ_{A_k}. More precisely, edges of Γ_{A_k} have two types of label; *input* and *output*. Labels we discussed above are input labels, are referred to simply as *labels*, and are elements of $X_1 \cup X_2$. Output labels are elements of $Z \cup \{1\}$. Edges of T_k all have output label 1; and each directed edge of Γ_{A_k} not in T_k has output label an element of Z. Each such edge corresponds uniquely to an element $h_{k,i}$ of the free generating set for H_k, and the edge corresponding to $h_{k,i}$ has output label $z_i \in Z$. Moreover, in this case $\phi_k(z_i) = h_{k,i}$. This means that if a word in H_k is given in terms of Y_k then its image under ϕ_k^{-1} can be read off from Γ_{A_k} (regarded as a transducer), by reading output labels.

Input.

- An inverse automaton Δ with alphabet Σ and start and final state 1, such that $\pi(L(\Delta)) = K$. Edges labelled with elements of X_k are said to have type k. Edges labelled with elements of Z are said to have type Z.

- Stallings automata Γ_{A_1} and Γ_{A_2} for H_1 and H_2, with output labels encoding ϕ_1 and ϕ_2, as above.
- For each $z \in Z$ the images $\phi_1(z)$ and $\phi_2(z)$ of as words in $Y_1^{\pm 1}$ and $Y_2^{\pm 1}$, respectively.

Output.

- An inverse automaton Δ'' (with alphabet Σ and start and final state 1, such that $\pi(L(\Delta'')) = K$).

Process.
Each of the following steps is executed for $k = 1$ and 2.

it:C1 C1 Construct Δ_k and the map ν. In more detail the steps are the following.

it:C1a (a) Remove all edges of type $X_{k'}$, where $k' = 1 - k$, from Δ; adding edges removed to a graph Δ_Z (which starts off empty, when $k = 1$, but is not reinitialised when k is incremented to 2).

it:C1b (b) A *shoot* is an edge incident to a leaf. Remove all shoots of type Z from Δ_k; adding edges removed to Δ_Z. Continue until there are no shoots of type Z in Δ_k.

it:C1c (c) Rename vertices of Δ_k: a vertex named v in Δ becomes (v, k) in Δ_k.

it:C1d (d) For each vertex (v, k) of Δ_k keep a record of the image of (v, k) in Δ (under ν). In practice, for each (v, k) keep a record $\nu\text{-im}(v, k)$ which is initially set to $\{v\}$.

it:C2 C2 (**Begin Modification 1**.) For all edges (α, z, β) of Δ_k, where $z \in Z$, add a path $(\alpha, \phi_k(z), \beta)$ to Δ_k. For later reference this version of Δ_k is called $\Delta'_{k,1}$.

it:C3 C3 Construct the Stallings folding of (each component of) Δ_k. The result is referred to as $\Delta_{k,1}$. Whenever two vertices (u, k) and (v, k) of Δ_k are identified, by the folding map, to a vertex (w, k) set $\nu\text{-im}(w, k) = \nu\text{-im}(u) \cup \nu\text{-im}(v)$. This process will be called **updating** ν, from now on.

it:C4 C4 (**Begin Modification 2**.) Construct $\mathcal{P}_k = \Delta_k \times \Gamma_{A_k}$.

it:C5 C5 Construct a spanning forest Υ_k of $\mathcal{P}_k$ and, simultaneously, the set L_{Υ_k} of words corresponding to paths in Υ_k from the root to each vertex. (A root of $\mathcal{P}_k$ must be chosen: to be explicit, assume the root is $((v, k), u)$, where u is minimal in some preassigned order on vertices of Γ_{A_k} and v is minimal in some chosen order of vertices of Δ.) We refer to these versions of $\mathcal{P}_k$ and Υ_k as $\mathcal{P}_{k,2}$ and $\Upsilon_{k,2}$ in the calculation of complexity below.

it:C6 C6 For each vertex α of Δ_k do the following. Let $\theta(\alpha)$ be the component of $\mathcal{P}_k$ containing $(\alpha, 1)$. For each vertex $(\beta, 1)$ of $\theta(\alpha)$ and for all paths p from $(\alpha, 1)$ to $(\beta, 1)$ add a path from α to β to the graph Δ_k, with label $\phi_k^{-1}(w)$, where w is the label of p. This version of Δ_k is referred to as $\Delta'_{k,2}$.

it:C7 C7 Construct the Stallings folding of Δ_k and update ν. This version of Δ_k is referred to as $\Delta_{k,2}$.

it:C8 C8 (**Begin Modification 3**.) Update $\mathcal{P}_k = \Delta_k \times \Gamma_{A_k}$ and Υ_k, by adding new isolated vertices if necessary. We refer to these versions of $\mathcal{P}_k$ and Υ_k as $\mathcal{P}_{k,3}$ and $\Upsilon_{k,3}$.

it:C9 C9 For all vertices of $\mathcal{P}_k$ of the form $(\alpha, 1)$ do the following. Let $\theta(\alpha)$ be the component of $\mathcal{P}_k$ containing $(\alpha, 1)$. For each edge e of $\theta(\alpha)\backslash\Upsilon_k$: if $e = ((\alpha_i, \beta_1), x, (\alpha_j, \beta_2))$, where α_i, α_j are vertices of Δ_k and β_1, β_2 are vertices of Γ_{A_k} and $x \in X_k$, add to Δ_k a path from α to α with label $\phi_k^{-1}(l_i x l_j)$, where l_i and l_j are the labels of paths in Υ_k from $(\alpha, 1)$ to (α_i, β_1) and to (a_j, β_2), respectively. This version of Δ_k is referred to as $\Delta'_{k,3}$.

it:C10 C10 Construct the Stallings folding of Δ_k and update ν. This version of Δ_k is referred to as $\Delta_{k,3}$.

it:C11 C11 (**Begin Modification 4**.) Update $\mathcal{P}_k = \Delta_k \times \Gamma_{A_k}$ and Υ_k, by adding new isolated vertices if necessary. We refer to these versions of $\mathcal{P}_k$ and Υ_k as $\mathcal{P}_{k,4}$ and $\Upsilon_{k,4}$.

it:C12 C12 For all vertices of $\mathcal{P}_k$ of the form $(\alpha, 1)$ do the following. Let $\theta(\alpha)$ be the component of $\mathcal{P}_k$ containing $(\alpha, 1)$. For all vertices (α_1, β) of $\theta(\alpha)$, with $\beta \neq 1$, let $b = w(\beta)$, the label of the path from 1 to β in T_k. If b is readable in the automaton $\mathcal{P}_k$ with start state $(\alpha, 1)$ and the final state, after reading b, is (α_1, β) there is nothing to do, for this vertex (α_1, β). Otherwise, let a be the label of the path from $(\alpha, 1)$ to (α_1, β) in Υ_k. Add to Δ_k a path from α to α_1 with label $(\phi_k^{-1}(ab^{-1}))b$. This version of Δ_k is referred to as $\Delta'_{k,4}$.

it:C13 C13 Construct the Stallings folding of Δ_k and update ν. This version of Δ_k is referred to as $\Delta_{k,4}$.

it:C14 C14 (**Begin Modification 5**.) Reconstruct $\mathcal{P}_k = \Delta_k \times \Gamma_{A_k}$, the spanning forest Υ_k and the set L_{Υ_k}. We refer to these versions of $\mathcal{P}_k$ and Υ_k as $\mathcal{P}_{k,5}$ and $\Upsilon_{k,5}$.

it:C15 C15 For all $\beta \in V(\Delta_k)$ do the following. For $(\varepsilon, \xi) \in P$, if both $b_1 = w(\varepsilon)$ and $b_2 = w(\xi)$ are readable in Δ_k, starting from β, and ending at vertices α_1 and α_2, then add to Δ_k a path from α_1 to α_2 with label $\phi_k^{-1}(b_1 c a_1^{-1}) a_1 a_2 [\phi_k^{-1}(b_2 c a_2^{-1})]^{-1}$, where (ε_0, ξ_0) is the $\sim$ representative of (ε, ξ), $a_1 = w(\varepsilon_0)$, $a_2 = w(\xi_0)$ and c is the connecting element of (ε, ξ). This version of Δ_k is referred to as $\Delta'_{k,4}$.

it:C16 C16 Construct the Stallings folding of Δ_k and update ν. This version of Δ_k is referred to as $\Delta_{k,5}$.

it:C17 C17 (**Begin Reassembly**.) For each vertex (u, k) of $\Delta_{1,5} \cup \Delta_{2,5}$, do the following. For each vertex (v, k') of $\Delta_{1,5} \cup \Delta_{2,5}$, not equal to (u, k), if $\nu\text{-im}((u, k)) \cap \nu\text{-im}((v, k')) \neq \emptyset$ then set $\nu\text{-im}((u, k)) = \nu\text{-im}((u, k)) \cup \nu\text{-im}((v, k'))$. For each edge e of $\Delta_{1,5} \cup \Delta_{2,5}$, if e has initial or terminal vertex equal to (v, k') then replace e with an edge having (u, k) as initial or terminal vertex, instead of (v, k'). Delete (v, k'), once no more such edges exist. Continue for as long as possible. The resulting graph is denoted Δ'_5.

it:C18 C18 For each vertex α of Δ_Z, if $\alpha \in \nu\text{-im}(u, k)$, for some vertex (u, k) of Δ'_5 then replace α with (u, k). (Note that, as we have completed step it:C17 C17, there is at most one such vertex of Δ'_5.) If α has been replaced by (u, k) then, for every edge e of Δ_Z, of the form (α, β) (or its reverse), replace e with the edge $((u, k), \beta)$ (or its reverse). Call the resulting graph Δ'_Z.

it:C19 C19 Form Δ'' from the union of Δ'_5 and Δ'_Z: by identifying vertices of Δ'_Z with vertices of Δ'_5 of the same name.

4.3 Summary of the Loop

Here the assumptions are as for Algorithim II.
Input.

- A tuple $k_1, \dots, k_s$ of elements of $F_1 *_{H_1=H_2} F_2$, in double coset normal form and the Stallings folding Γ_K of this tuple as a set of words in $F_1 * F_2 * F(Z)$.

Output.

- An inverse automaton Ψ, with alphabet Σ and start and final state 1, such that $\pi(L(\Psi)) = K$ and Ψ accepts the double coset normal form of every element of K.

Process.

D1 Set $n = 0$ and $\Delta_{(0)} = \Gamma_K$.

it:loop2 D2 Input $\Delta_{(n)}$ to Algorithm II. Call the output $\Delta''_{(n+1)}$.

it:loop3 D3 Fold $\Delta''_{(n+1)}$ and call the result $\Delta_{(n+1)}$.

D4 If the number of X_1 and X_2 components of $\Delta_{(n)}$ and $\Delta_{(n+1)}$ are the same, output $\Delta_{(n+1)}$ and halt. Otherwise, add 1 to n and go to Step D2.

4.4 Generalised folding example

ex:K **Example 4.1.** Let $F_1 = \mathbb{F}(x_1, x_2, x_3)$ and $H_1 = \langle h_1, h_2, h_3\rangle$, as in Example 2.7, and let $F_2 = \mathbb{F}(y_1, y_2, y_3, y_4)$ and $H_2 = \langle h'_1, h'_2, h'_3\rangle$, as in Example 2.8. Let f_1, f_2 and f'_1 be the elements defined in these examples and let $G = F_1 *_{H_1=H_2} F_2$.

Let

$$f_3 = x_2x_3^{-1}x_1,\ f_4 = x_1^4x_2x_3x_1^{-1}x_2^{-1} \text{ and } f_5 = x_1x_2x_3x_2x_3^{-2}x_2^{-1},$$

and let

$$f'_2 = y_3y_4y_2^{-1}y_1y_3.$$

Using Algorithm I, the double coset representatives of these elements are

$$\begin{aligned} f_3 &= x_2x_3^{-1}x_1,\\ f_4 &= h_1h_3x_1^{-1}x_2^{-1},\\ f_5 &= h_3h_2^{-2} \text{ and}\\ f'_2 &= h'_2y_2^{-1}y_1y_3. \end{aligned}$$

Let K be the subgroup of G generated by $k_1 = f_1f'_1f_2$, $k_2 = f_3f'_2f_4$ and $k_3 = f_5$. In Example 3.3 we found the double-coset normal form of $k_1 = g = f_1f'_1f_2$, namely

$$k_1 = z_2^2z_1x_1x_3x_1z_1^{-1}z_2y_1y_2^{-1}z_1^{-1}z_2^{-1}z_3^{-1}z_2^{-1}x_1^{-1}.$$

The double coset normal forms of k_2 and k_3 are

$$k_2 = x_2 x_3^{-1} x_1 z_2 y_2^{-1} y_1 y_3 z_1 z_3 x_1^{-1} x_2^{-1}$$

and

$$k_3 = z_3 z_2^{-2}.$$

We form the flower automaton of K and then its (classical) Stallings folding, which is shown in Figure 4.3. There is one X_1 component Θ of this graph, shown in Figure 4.4 and two X_2 components, Θ' and Θ'', shown in Figure 4.5. Each of these components is input to the loop, in turn.

First consider the X_1 component Θ. Modification 1 results in the graph of Figure 4.6. To perform the next stages of the modification process the graph $\Theta_1 \times \Gamma_{A_1}$ is constructed; where Γ_{A_1} is the graph of Example 2.7, shown in Figure 2.1a. The product graph $\mathcal{P}_1$ is shown in Figure 4.7. The spanning forest chosen for $\mathcal{P}_1$ is the one obtained by removing the edge joining vertices $(13, 1)$ and $(30, 5)$. On examination of $\mathcal{P}_1$ there is one path to add: corresponding to the path in $\mathcal{P}_1$, from $(15, 1)$ to $(31, 1)$, labelled x_1^{-3}. Thus a path is added to Θ_1, from the vertex 15 to the vertex 31, with label z_1^{-1}. There are no other paths to add which do not already exist in Θ_1. The result is the graph Θ_2, shown in Figure 4.8. The next step is to construct $\mathcal{P}_2 = \Theta_2 \times \Gamma_{A_1}$, but as there are no X_1 edges in Θ_2 that are not in Θ_1, it follows that $\mathcal{P}_2$ is the union of $\mathcal{P}_1$ and some isolated vertices, and we may use $\mathcal{P}_1$ instead of $\mathcal{P}_2$. In this instance there is nothing to add at this stage and $\Theta_3 = \Theta_2$.

In formation of Θ_4 paths are added to Θ_3, corresponding to pairs of vertices (γ, δ) of $\mathcal{P}_3 (= \mathcal{P}_1)$, which are not covered by $\mathcal{P}_3$. In the table of Figure 4.1 the paths to be added are listed: each row corresponds to such a pair (γ, δ). The label of the path from $\gamma = (\alpha_i, 1)$ to $\delta = (\alpha_j, \beta)$ in $\mathcal{P}_3$ is a, the label of the path from 1 to β in Θ_k is b, and the word $\phi_k^{-1}(ab^{-1})$ is w. For each line of the table a path, from α_i to α_j, with label wb, is added to Θ_3. (These are not the only pairs of vertices which are not covered: however, the other pairs do not result in any further paths being added.) After adding these paths to Θ_3 the graph is folded to produce the graph Θ_4, shown in Figure 4.9.

To form Θ_5 we first construct $\mathcal{P}_4 = \Theta_4 \times \Gamma_{A_k}$, which is shown in Figure 4.10, and choose a spanning subforest Υ_4 of $\mathcal{P}_4$. In this example this is done by removing the edge joining vertices $(13, 1)$ and $(30, 5)$ from $\mathcal{P}_4$. A path will be added to Θ_4 whenever a pair (ε, ξ) in the set P of non-diagonal pairs of vertices of Γ_{A_k} is found, satisfying the conditions given in Modification 5: namely that

- (ε, ξ) is not a $\sim$ representative;

γ	δ	a	b	w
$(25,1)$	$(13,5)$	x_2x_3	x_2	z_2
$(29,1)$	$(25,2)$	$x_3^{-1}x_2^{-1}x_1$	x_1^{-1}	$z_3^{-1}z_1$
$(32,1)$	$(15,2)$	x_1^2	x_1^{-1}	z_1
$(32,1)$	$(13,3)$	x_1^{-2}	x_1	z_1^{-1}
$(32,1)$	$(30,4)$	$x_1^{-2}x_2$	x_3^{-1}	$z_1^{-1}z_3$
$(1,1)$	$(3,5)$	$x_2x_3^{-1}$	x_2	z_2^{-1}
$(30,1)$	$(12,3)$	$x_3^{-1}x_2^{-1}$	x_1	z_3^{-1}

tab:T4 Figure 4.1: Paths to be added to form Θ_4

- there are paths p_1 and p_2 in Υ_4 from vertices $(\alpha_i, 1)$ to (β, ε) and from $(\alpha_j, 1)$ to (β, ξ), for some $\alpha_i, \alpha_j, \beta \in V(\Theta_4)$, and
- the labels of p_1 and p_2 are the same as the labels of the paths, from 1 to ε, and 1 to ξ, respectively, in T_k.

If such a pair (ε, ξ) is found then let (ε_0, ξ_0) be its $\sim$ representative, let c be the connecting element and let a_1 and a_2 be the labels of the paths, from 1 to ε_0, and 1 to ξ_0, in T_k. Then the label of any corresponding path added to Θ_4 is $w = \phi_k^{-1}(b_1ca_1^{-1})a_1a_2^{-1}\phi_k^{-1}(a_2cb_2^{-1})$. The table of Figure tab:T5 4.2 shows pairs of vertices $(\alpha_i, 1)$, (β, ε) and $(a_l, 1)$, (β, ξ), of $\mathcal{P}_4$, for which the conditions above are satisfied. In each case the new path is from α_i to α_l. There are only two pairs (ε, ξ) which arise: namely $(4,5)$ and $(2,3)$, and the labels of the new paths added are $z_3^{-1}x_1$ and z_1x_1, respectively. After adding these paths to Θ_4 the graph is folded to produce the graph Θ_5, shown in Figure fig:K_f5 4.11.

Next consider the X_2 component Θ'. There is nothing to add to form Θ'_1. The graph $\mathcal{P}'_1 = \Theta'_1 \times \Gamma_{A_2}$ is shown in Figure fig:K_i1-x-g 4.12. There is nothing to add at the next two stages so $\Theta'_1 = \Theta'_2 = \Theta'_3$, so $\mathcal{P}'_3 = \mathcal{P}'_1$. There is one pair of vertices not covered by $\mathcal{P}_3$: namely $\gamma = (6,1)$ and $\delta = (5,6)$. Thus a path is added to Θ'_3, from 6 to 5, with label $z_1y_2^{-1}$, as shown in Figure fig:K_i4 4.13a. The graph $\mathcal{P}'_4 = \Theta'_4 \times \Gamma_{A_2}$ is shown in Figure fig:K_i4-x-g 4.14. The paths from $(5,1)$ to $(7,5)$ and from $(6,1)$ to $(7,3)$, in $\mathcal{P}_4$, give rise to a path to be added to Θ'_4. However this path has label $y_2z_1^{-1}$, and joins 5 to 6, so already belongs to Θ'_4. Hence $\Theta'_4 = \Theta'_5$.

$(\alpha_i, 1)$	(β, ε)	$(\alpha_l, 1)$	(β, ξ)
$(2,1)$	$(3,4)$	$(43,1)$	$(3,5)$
$(13,1)$	$(34,4)$	$(25,1)$	$(34,5)$
$(14,1)$	$(32,2)$	$(31,1)$	$(32,3)$
$(15,1)$	$(14,2)$	$(32,1)$	$(14,3)$
$(25,1)$	$(1,2)$	$(36,1)$	$(1,3)$
$(31,1)$	$(13,2)$	$(40,1)$	$(13,3)$

tab:T5 Figure 4.2: Paths to be added to form Θ_5

Finally consider the X_2 component Θ''. A similar analysis, details of which are omitted, shows that Θ''_5 is the graph shown in Figure fig:K_j4 4.13b.

The folding of the reassembled graph Ψ is shown in Figure fig:out 4.15. As this graph has the same number of X_1 and X_2 components as the original, the algorithm halts after one iteration of the Loop, outputting Ψ.

5 Proofs of main results

lem:nfcomp **Lemma 5.1.** *Let α and β be boundary vertices of an X_k component Θ of an inverse automaton Δ, with alphabet Σ. Then the following hold.*

it:nfcomp1 1. $\pi(L(\Theta, \alpha, \beta)) = \pi(L(\Theta_5, \theta(\alpha), \theta(\beta)))$.

it:nfcomp2 2. *Let w be a word which is accepted by the automaton (Θ, α, β). Then the normal form of w is accepted by $(\Theta_5, \theta(\alpha), \theta(\beta))$.*

Proof. it:nfcomp1 1. At each stage of the modification process of Algorithm II the image under π of the language accepted by $(\Theta_i, \alpha, \beta)$ was preserved.

it:nfcomp2 2. We may assume that $w \in \mathbb{F}(X_k)$ (given the construction of Θ_1). Let w have normal form $h_1 s h_2^{-1}$, where $s \in S_k$ and $h_i \in \mathbb{F}(Z)$. There are two cases to consider, depending on whether s is a representative of type 1 or type 2. First consider the case where s is of type 1, say $s = a_1 e a_2^{-1}$, where a_1 is a maximal L_{Q_k}-prefix and an L_{T_k}-prefix of s and a_2 is a maximal L_{Q_k}-prefix and an L_{T_k}-prefix of $a_2 \circ e^{-1}$. Then, as the normal form is the result of applying ϕ_k to the output of Algorithm I, there are words g_1, g_2, b_1 and $b_2 \in \mathbb{F}(X_k)$ such that $w = g_1 \circ b_1 \circ e \circ b_2^{-1} \circ g_2^{-1}$, $g_i \in H_k$, b_1 is a maximal

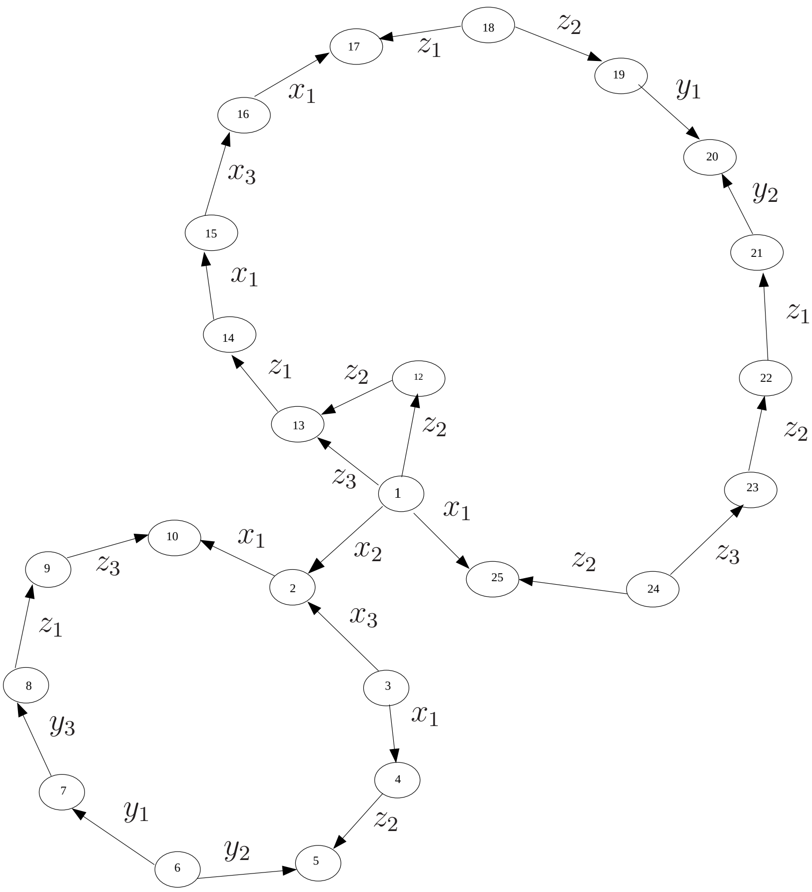


fig:Kflower

Figure 4.3: The folded flower automaton of K

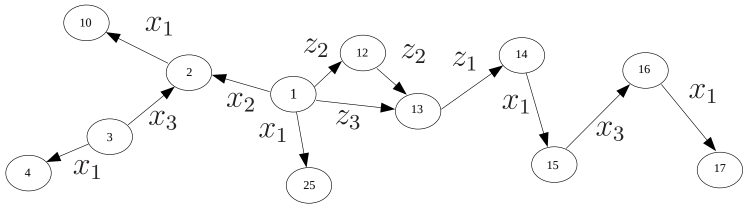


fig:KX

Figure 4.4: X_1 component Θ

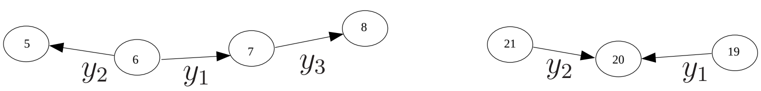


fig:KY1 (a) Θ' fig:KY2 (b) Θ''

fig:KY

Figure 4.5: X_2 components

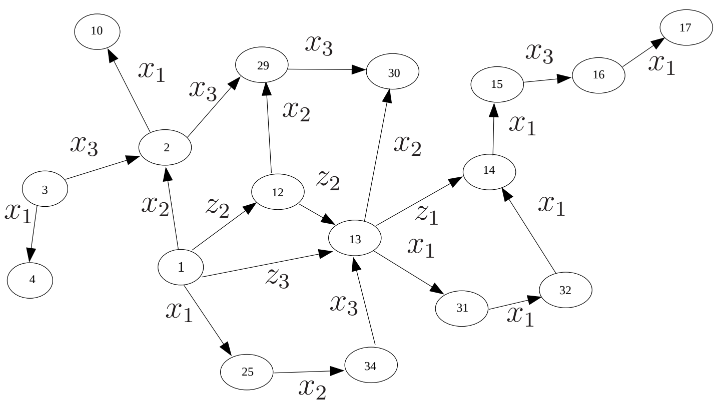
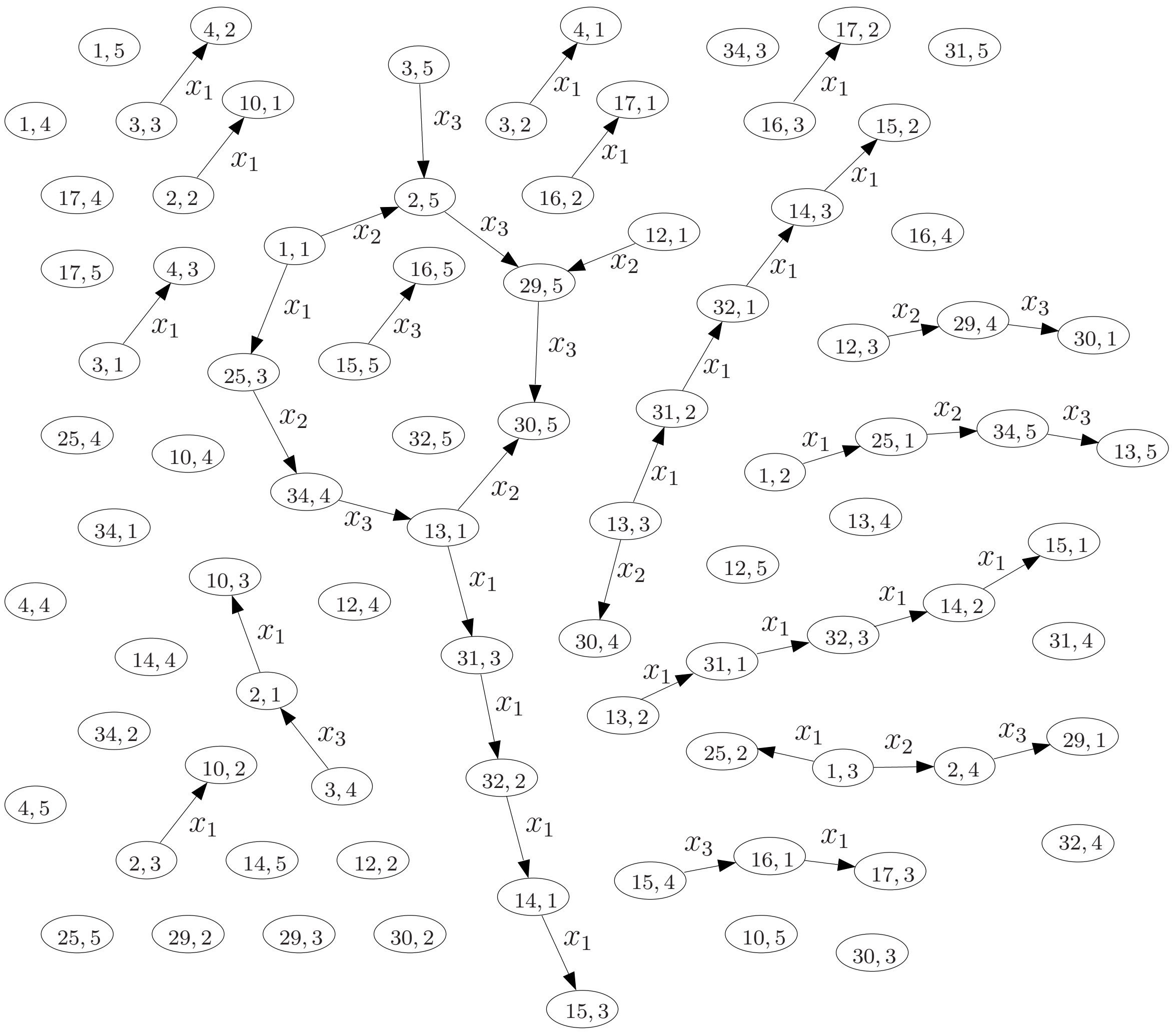


fig:K_f1

Figure 4.6: X_1 component Θ_1

L_Q-prefix of $b_1 \circ e \circ b_2^{-1} \circ g_2^{-1}$, b_2 is a maximal L_Q-prefix of $b_2 \circ e^{-1}$, $e \neq 1$, $a_i = w(\tau(b_i))$ and $\phi_k(h_i) = g_i b_i a_i^{-1}$, $i = 1, 2$.

Recall that we refer to the images of vertices of Θ in Θ_i as "vertices of Θ", for $i = 1, \ldots, 5$. Since $g_i \in H_k$ and w is accepted by Θ_1 there is a path labelled g_1 from α to a vertex α_1 of Θ and a path labelled g_2 from β to a vertex β_1 of Θ. Therefore, in $\mathcal{P}$, there are paths p_1 from $(\alpha, 1)$ to $(\alpha_1, 1)$ labelled g_1, and p_2 from $(\beta, 1)$ to $(\beta_1, 1)$ labelled g_2. (See Figure fig:nf-1 5.1.) Now p_1 may be written as a concatenation of paths $p_1 = o_0 e_1 \cdots e_l o_l$, where o_i is a simple path in Υ and e_i is an edge of $\mathcal{P}$ which does not belong to Υ. Let e_i have initial and terminal vertices γ_i and δ_i, and let L_i and R_i be the simple paths in Υ from $(\alpha, 1)$ to γ_i and from δ_i to $(\alpha, 1)$, respectively. In addition let L_{l+1} be the path in Υ from $(\alpha, 1)$ to $(\alpha_1, 1)$. Then $L_1 = o_0$ and, for $i = 1, \ldots, l$, $L_{i+1} = R_i^{-1} o_i$. (See Figure fig:LeR 5.2.) Moreover, for $i = 1, \ldots, l$, the path $L_i e_i R_i$ is a closed path in $\mathcal{P}$, based at $(\alpha, 1)$, containing exactly one edge, e_i, which is not in Υ. Thus the label of $L_i e_i R_i$ is $v_i \in H_k$ and $\phi_k^{-1}(v_i)$ is the label of a closed path in Θ_3, based at α. (All such paths were added in the construction of Θ_3 from Θ_2.) Also, the path L_{l+1}, from $(\alpha, 1)$ to $(\alpha_1, 1)$, has label $v_{l+1} \in H$, and by construction of Θ_2 there is a path from α to α_1 in Θ_2 with label $\phi_k^{-1}(v_{l+1})$. The path p_1 is the result of reducing (deleting adjacent edges e, e^{-1}) the path $o_0 e_1 (R_1 R_1^{-1}) o_1 \cdots e_l (R_l R_l^{-1}) o_l$, which is equal to the path



`fig:K_f1-x-g`

Figure 4.7: $\mathcal{P}_1 = \Theta_1 \times \Gamma_{A_1}$

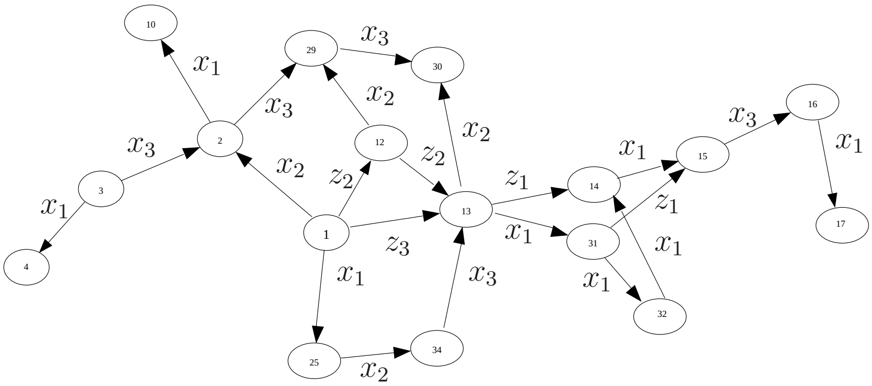


fig:K_f2

Figure 4.8: X_1 component Θ_2

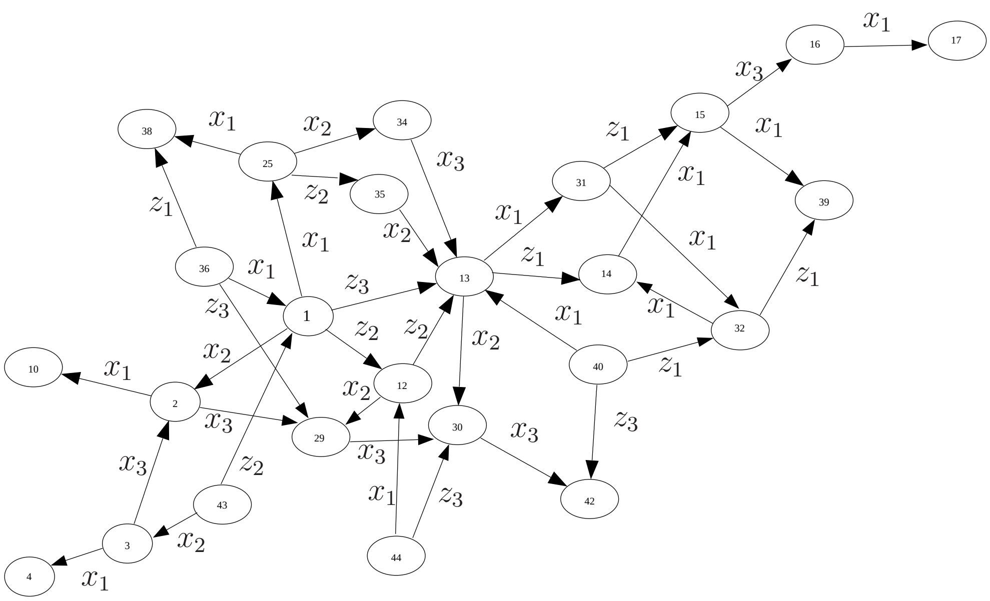


fig:K_f4

Figure 4.9: X_1 component Θ_4

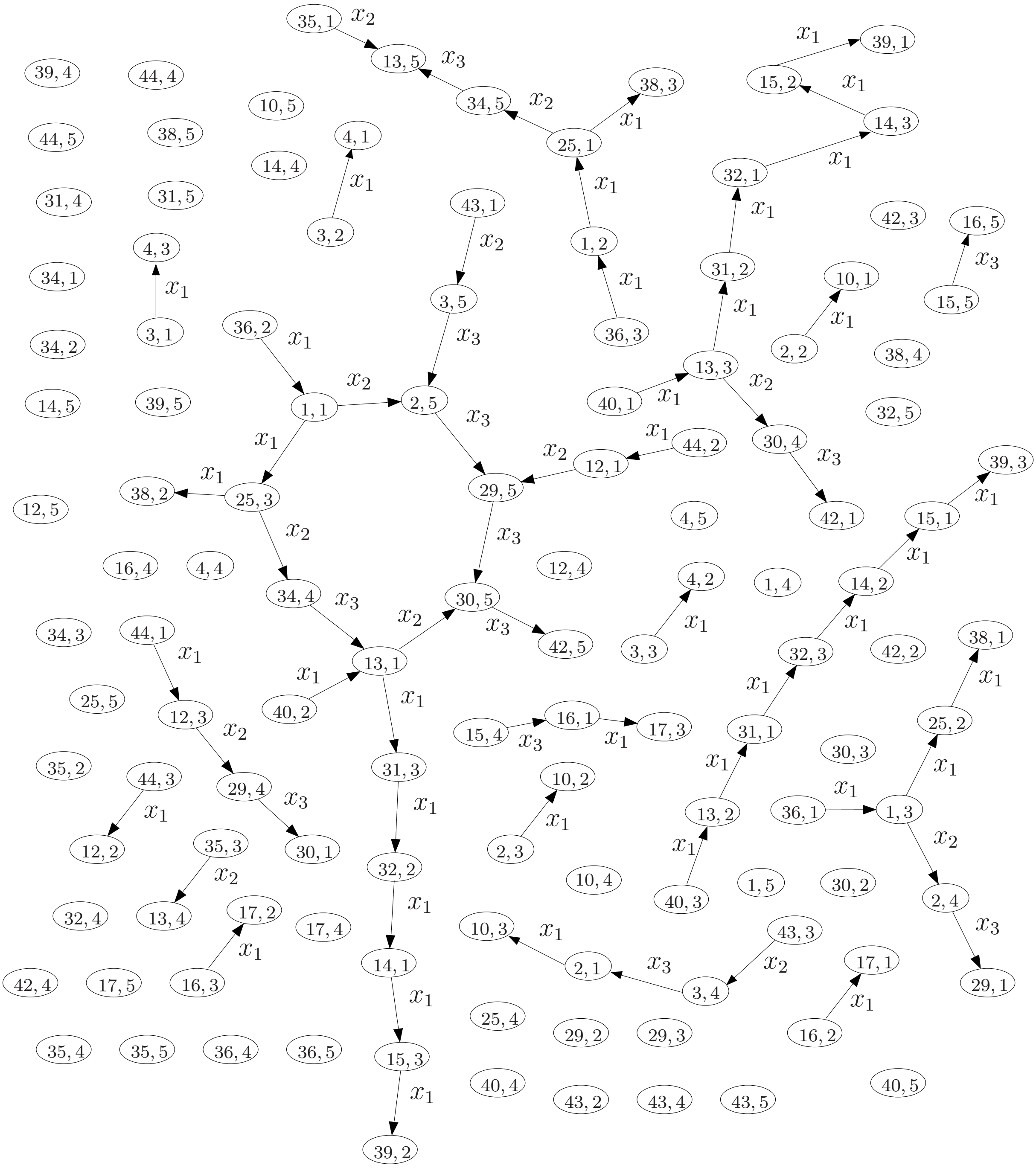


fig:K_f4-x-g

Figure 4.10: $\mathcal{P}_4 = \Theta_4 \times \Gamma_{A_1}$

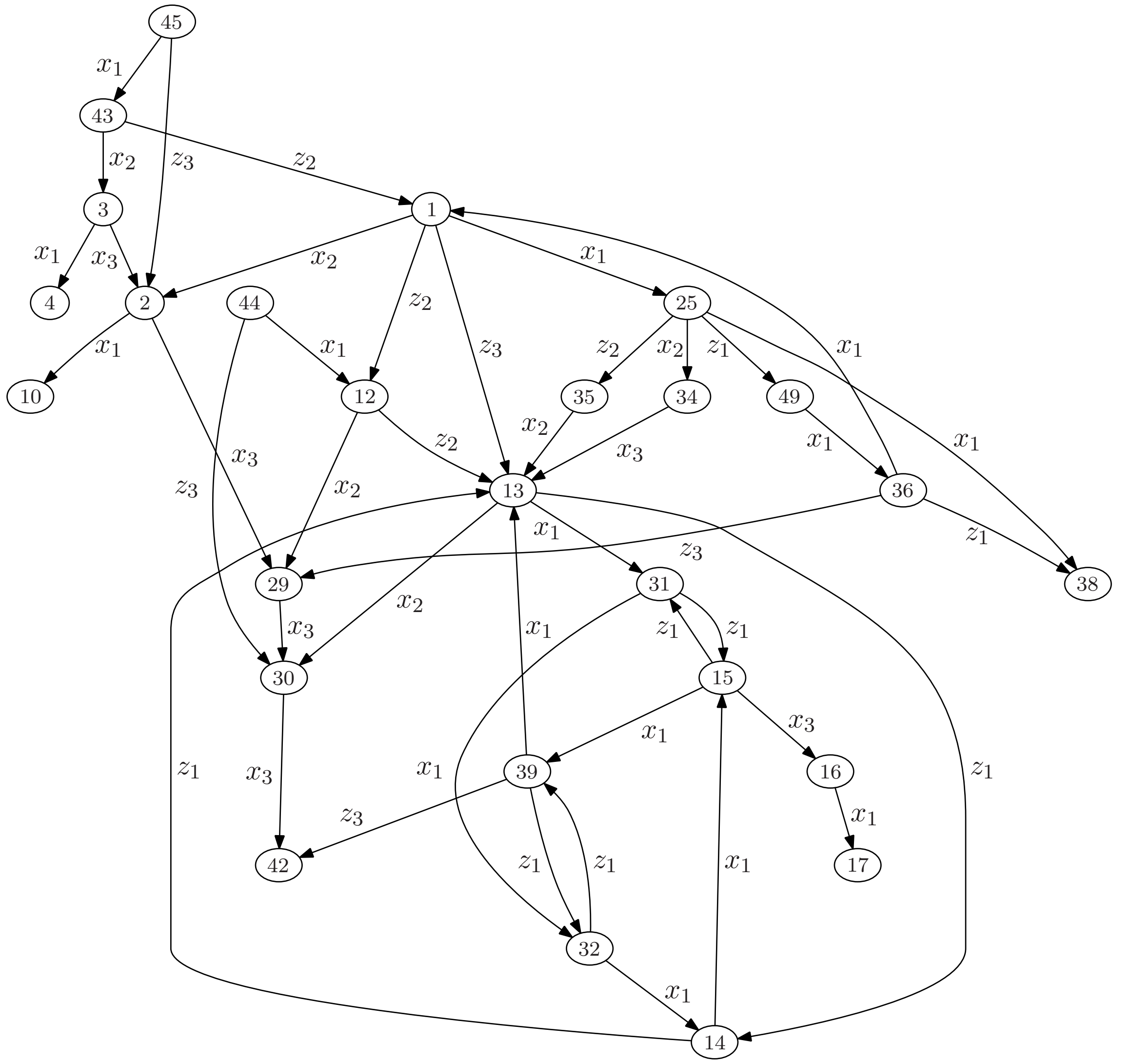


fig:K_f5

Figure 4.11: Θ_5

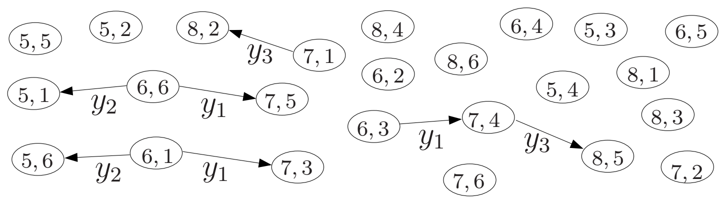


fig:K_i1-x-g

Figure 4.12: $\mathcal{P}'_1 = \Theta'_1 \times \Gamma_{A_2}$

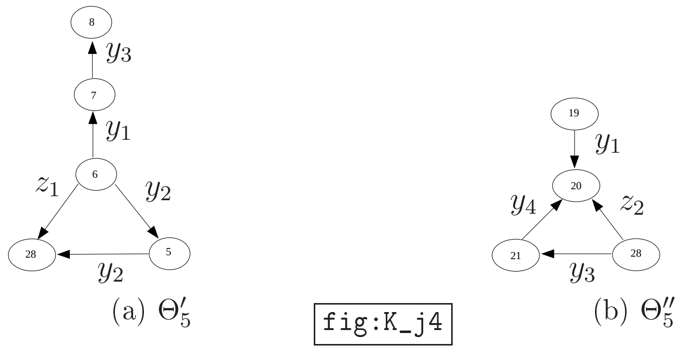


fig:K_i4

(a) Θ'_5

fig:K_j4

(b) Θ''_5

fig:KY5

Figure 4.13: X_2 components

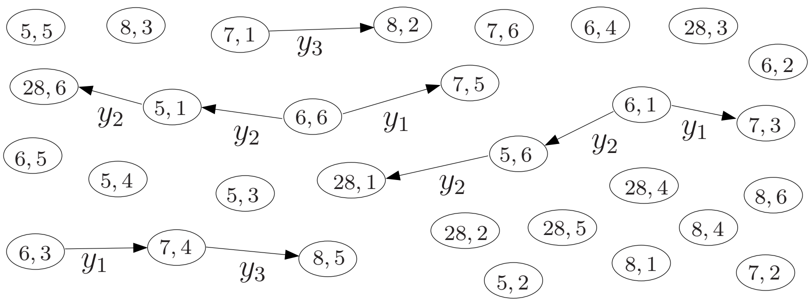


fig:K_i4-x-g

Figure 4.14: $\mathcal{P}'_4 = \Theta'_4 \times \Gamma_{A_2}$

$L_1 e_1 R_1 \cdots L_l e_l R_l L_{l+1}$. Hence the label g_1 of p_1 is the result of reducing the word $v_1 \cdots v_l v_{l+1}$ and $\phi_k^{-1}(g_1)$ is the result of reducing $\phi_k^{-1}(v_1) \cdots \phi_k^{-1}(v_{l+1})$. As each $\phi_k^{-1}(v_i)$ is readable in Θ_3 and Θ_3 is folded this means that $\phi_k^{-1}(g_1)$ is the label of a path, from α to α_1, in Θ_3. Similarly, $\phi_k^{-1}(g_2)$ is the label of a path from β to β_1 in Θ_3.

The word b_1 is readable by Γ_{A_k} and there is a path with label b_1 in Θ_1 from α_1 to some vertex α_2 of Θ. Hence b_1 is the label of a path in $\mathcal{P}$ from $(\alpha_1, 1)$ to $(\alpha_2, \varepsilon_1)$, for some vertex ε_1 of Γ_{A_k}. By definition $a_1 = w(\tau(b_1))$ is the label of a path in T_k from 1 to ε_1. Let b'_1 be the label of the simple path p'_1 in Υ from $(\alpha_1, 1)$ to $(\alpha_2, \varepsilon_1)$ (see Figure fig:nf-1 5.1). By construction Θ_4 contains a path from α_1 to α_2 with label $h'_1 a_1$, where $h'_1 = \phi_k^{-1}(b'_1 a_1^{-1}) \in \mathbb{F}(Z)$. (If $\mathcal{P}$ covers the pair $(\alpha_1, 1)$, $(\alpha_2, \varepsilon_1)$ then $b'_1 = a_1$ and h'_1 is the empty word, while there exists a path with label a_1 from α_1 to α_2 in Θ_3.) Now $b_1(b'_1)^{-1} \in H_k$ and is the label of a closed path in $\mathcal{P}$ based at $(\alpha_1, 1)$; so by the previous part of proof, Θ_3 contains a closed path, based at α_1, with label $w'_1 = \phi_k^{-1}(b_1(b'_1)^{-1})$. Hence, as it is folded, Θ_4 contains a path, from α to α_2, with label the reduced word obtained from the product $\phi_k^{-1}(g_1) w'_1 h'_1 a_1$, that is $\phi_k^{-1}(g_1 b_1 a_1^{-1}) a_1 = h_1 a_1$. Similarly, Θ_1 contains a path labelled b_2 from β_1 to some vertex β_2 of Θ, and Θ_4 contains a path from β to β_2 with label $h_2 a_2$.

As w is the label of a path in Θ_1 there is a path from α_2 to β_2 in Θ_4 with label e. Therefore, in this case, Θ_4 contains a path from α to β which has label the normal form of w.

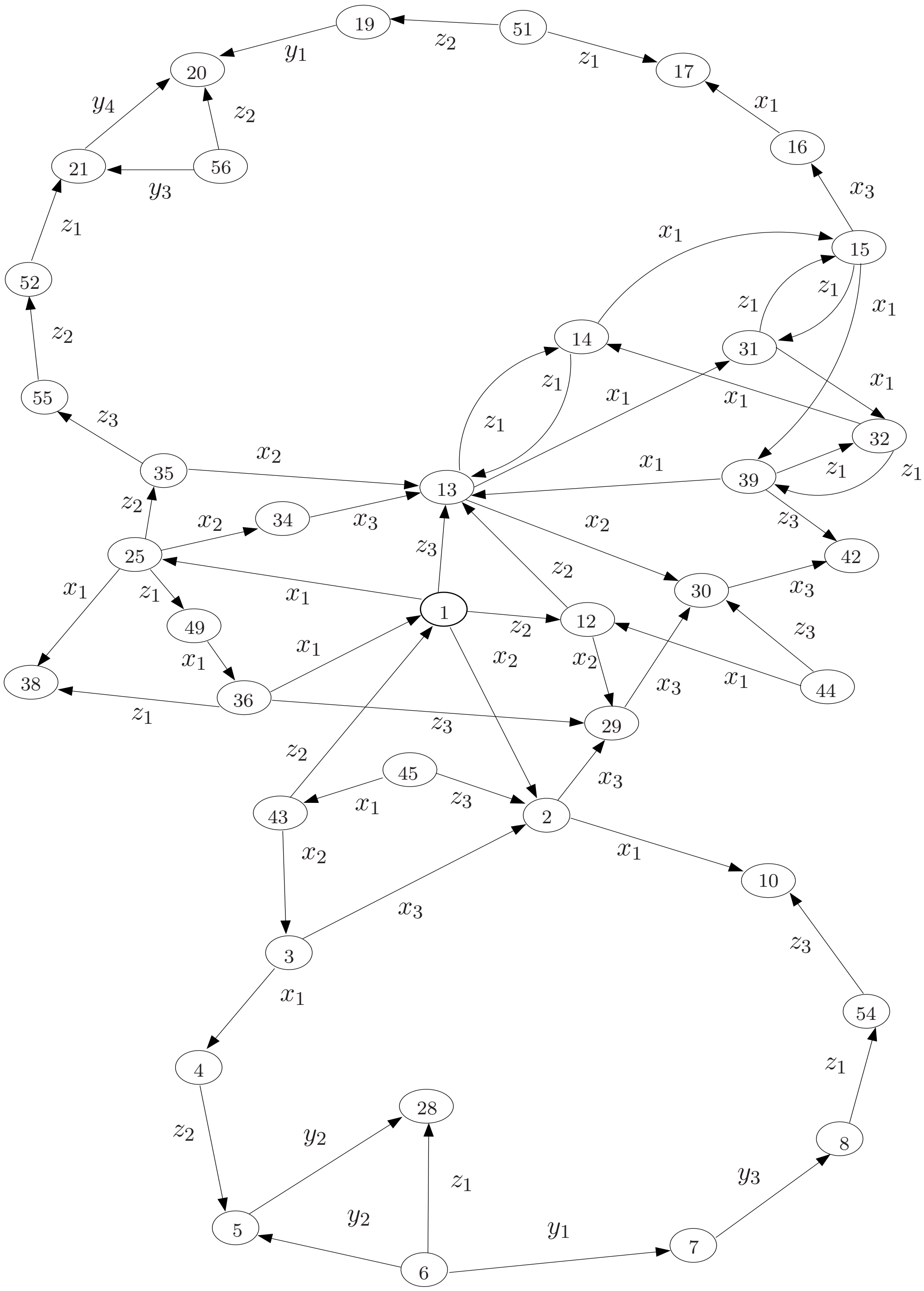


fig:out

Figure 4.15: Ψ

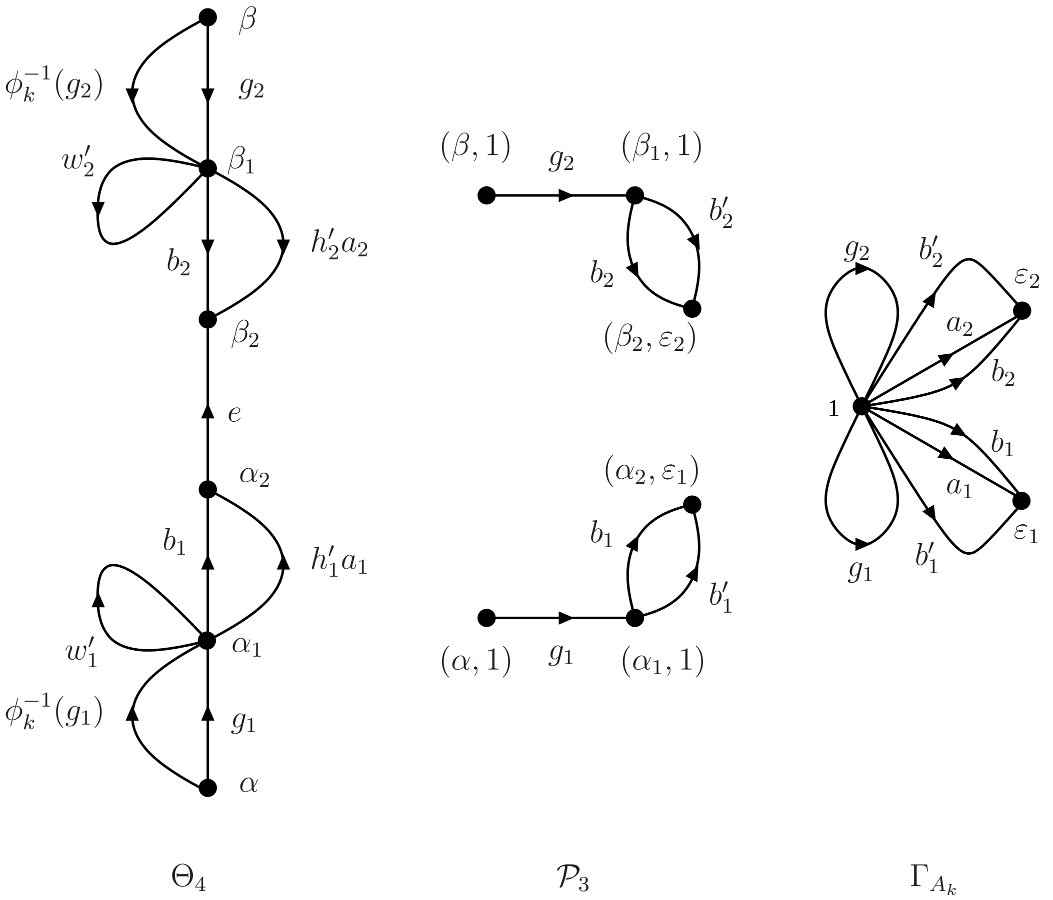


fig:nf-1

Figure 5.1: Normal form: type 1.

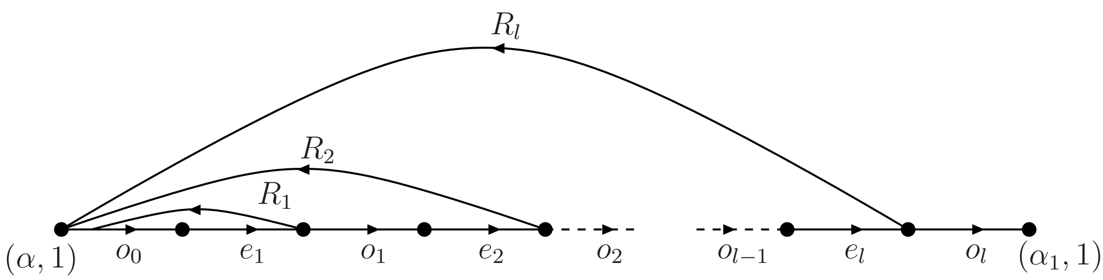


fig:LeR

Figure 5.2: The path p_1.

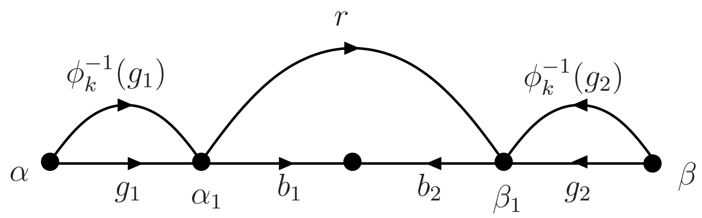


fig:nf-2 Figure 5.3: Normal form: type 2, $r = \phi_k^{-1}(b_1ca_1^{-1})a_1a_2^{-1}\phi_k^{-1}(a_2c^{-1}b_2^{-1})$.

In the second case, let w have normal form $h_1a_1a_2^{-1}h_2^{-1}$, where h_1 and h_2 are in $\mathbb{F}(Z)$, a_1 and a_2 are labels of simple paths in the subtree T_k and $s = a_1a_2^{-1}$ is a double coset representative of type 2. Then there are words g_1, g_2, b_1, b_2, c in $\mathbb{F}(X_k)$, such that $w = g_1 \circ b_1 \circ b_2^{-1} \circ g_2^{-1}$, $g_i \in H_k$, b_1 is a maximal L_Q-prefix of $b_1 \circ b_2^{-1} \circ g_2^{-1}$, b_2 in L_Q, $c = c(\tau(b_1), \tau(b_2))$ and $a_i = w(v_i)$, where (v_1, v_2) is the $\sim$ representative of $(\tau(b_1), \tau(b_2))$.

As in the first case there are paths in Θ_3 labelled $\phi_k^{-1}(g_1)$ and $\phi_k^{-1}(g_2)$ from α to α_1 and β to β_1, respectively. (See Figure 5.3.) As w is readable in Θ there is a path labelled $b_1b_2^{-1}$, from α_1 to β_1, in Θ_3. By construction of Θ_5 there is also a path labelled $r = \phi_k^{-1}(b_1ca_1^{-1})a_1a_2^{-1}\phi_k^{-1}(a_2c^{-1}b_2^{-1})$ from α_1 to β_1. As Θ_5 is folded $\phi_k^{-1}(g_1b_1ca_1^{-1})a_1a_2^{-1}\phi_k^{-1}(a_2c^{-1}b_2^{-1}g_2)$ is readable in Θ_5. Since the latter is in normal form we have $h_1 = \phi_k^{-1}(g_1b_1ca_1^{-1})$ and $h_2 = \phi_k^{-1}(g_2b_2ca_2^{-1})$, as required. $\square$

lem:idverts **Lemma 5.2.** *Let $[\cdot]$ be the equivalence relation on $V(\Delta_5')$ above. If x and y are vertices of Δ', such that $[\theta(x)] = [\theta(y)]$ in Δ'', then there exists a path in Δ, from $\nu(x)$ to $\nu(y)$, with label w, such that $\pi(w) = 1$.*

Proof. Let $\alpha = \theta(x)$ and $\beta = \theta(y)$. As $[\alpha] = [\beta]$ there exists a sequence of vertices $\gamma_0, \dots, \gamma_m$ of Δ_5' such that $\alpha = \gamma_0$, $b = \gamma_m$ and

$$\nu(\theta^{-1}(\gamma_i)) \cap \nu(\theta^{-1}(\gamma_{i+1})) \neq \emptyset,$$

for $i = 0, \dots, m-1$. Suppose first that $m = 0$. Then $\theta(x) = \alpha = \beta = \theta(y)$, so x and y both belong to some X_k component Θ of Δ'. As $\pi(L(\Theta, x, y)) = \pi(L(\Theta_5, \alpha, \beta) = \pi(L(\Theta_5, \alpha, \alpha))$, there is a path in Θ, from x to y, with label w, such that $\pi(w) = 1$. Θ is embedded in Δ by the map ν, so the same holds for $\nu(x)$ and $\nu(y)$ in Δ.

Now assume that $m > 0$ and that the result holds in all cases where the sequence of γ_i's above has length at most m. Let $a \in \nu(\theta^{-1}(\gamma_{m-1})) \cap \nu(\theta^{-1}(\gamma_m))$, and let b, g be vertices of Δ' such that $\nu(b) = \nu(g) = a$, $\theta(b) = g_m = \beta$ and $\theta(g) = \gamma_{m-1}$. As $[\alpha] = [\gamma_1] = \cdots = [\gamma_{m-1}] = [\gamma_m]$, it follows from the inductive hypothesis that there is a path in Δ, from x to a, with label w_1, such that $\pi(w_1) = 1$. Moreover, $\theta(b) = \beta = \theta(y)$, so it follows from the case $m = 0$, that there exists a path in Δ, from a to y, with label w_2, such that $\pi(w_2) = 1$. Concatenating these paths we obtain the required result. □

lem:dcfold **Lemma 5.3.** *Let Δ be an inverse automaton, with alphabet Σ and start state s, let Δ'' and Ψ be the dc-resolution and dc-folding of Δ, respectively, and let $\hat{\rho} : \Delta \to \Psi$ be the dc-folding morphism.*

Then the following hold.

it:dcfold1 1. *Ψ is an inverse automaton, with unique start and final state $\sigma = \hat{\rho}(s)$, and has no more X_1 and X_2 components than Δ.*

it:dcfold2 2. *$\pi(L(\Psi, \sigma)) = \pi(L(\Delta, s))$.*

Proof. 1. The automaton Ψ is involutive and folded by definition. Its root is the image of the root $\rho(s)$ of Δ'' under the folding morphism: that is $\hat{\rho}(s)$. As Δ is connected so is Δ'' and therefore also Ψ. Hence Ψ is inverse. The graphs Δ and Δ'' have the same number of X_k components. Folding to form Ψ cannot increase this number, so the final statement of 1 follows.

2. Let w be a word accepted by $\Psi = (\Psi, \sigma)$. Since Ψ is formed by a sequence of foldings of Δ'', there is a word v such that v is accepted by Δ'' (with root $\rho(s)$) and $\pi(v) = \pi(w)$. Thus v is the label of a path q_i in Δ'', from $\rho(s)$ to $\rho(s)$, and $v = v_1 \cdots v_n$, where each factor v_i is accepted by a connected component Ξ_i of Δ'_5; and no two consecutive factors are accepted by the same component. From Lemma 5.1.1 it follows that there are paths $p_1, \ldots, p_n$, in Δ', with labels $u_1, \ldots, u_n$, such that $\pi(u_i) = \pi(v_i)$ and p_i is a path in the component $C_i = \theta^{-1}(\Xi_i)$ of Δ'. Let the initial and terminal vertices of p_i be x_i and y_i. As ν restricted to C_i is an embedding, there is a path in Δ, from $\nu(x_i)$ to $\nu(y_i)$ with label u_i, for $i = 1, \ldots, n$.

Moreover, as v is a path in Δ'', based at $\rho(s)$, we have

$$[\theta(x_1)] = [\theta(y_n)] = \rho(s) \text{ and } [\theta(y_i)] = [\theta(x_{i+1})],$$

for $i = 1, \ldots, n-1$. From Lemma 5.2, there exist paths in Δ,

- from s to $\nu(x_1)$, with label w_0;
- from $\nu(y_n)$ to s, with label w_n and
- from $\nu(y_i)$ to $\nu(x_{i+1})$, with label w_i, for $i = 1, \ldots, n-1$,

such that $\pi(w_i) = 1$, for $i = 1, \ldots, n$. Thus $l = w_0 u_1 \cdots w_{n-1} u_n w_n$ is the label of a path in Δ, based at s, such that $\pi(l) = \pi(u_0 \cdots u_n) = \pi(v) = \pi(w)$. Therefore $\pi(L(\Psi, \sigma)) \subseteq \pi(L(\Delta, s))$. The reverse inclusion follows from the existence of the morphism $\hat{\rho}$ from Δ to Ψ. □

If w is a word in $\mathbb{F}(X_1 \cup X_2 \cup Z)$ and $w = w_1 \cdots w_n$, where $w_i \in \mathbb{F}(X_{k_i} \cup Z)$, with $k_i \neq k_{i+1}$ and n minimal, then say that each w_i is a *maximal* X_{k_i} *factor* of w.

lem:loopstop **Lemma 5.4.** *Let Δ_0 be an inverse automaton with alphabet Σ.*

it:loopstop1 *1. If Δ_0 is input to the loop then the loop halts after finitely many iterations and outputs an inverse automaton Δ_n.*

it:loopstop2 *2. Let Δ_n be the output from the loop and let w be a word accepted by Δ_0. If $w = a \circ b \circ c$, where b is a maximal X_k factor of w then Δ_n accepts the word avc, where v is the normal form of b.*

Proof. it:loopstop1 1. The number of X_k components is not increased by Steps 1 and 2, so at some point the loop must halt. it:loopstop2 2. The word b is accepted by an X_k component of Δ_0. By construction, the word v is accepted by Δ_1, and hence by Δ_n. It follows that Δ_n accepts avc. □

lem:nfacc **Lemma 5.5.** *Let Δ_0 be an inverse automaton with alphabet Σ and let Δ_n be the output from the loop, with input Δ_0. If w is accepted by Δ_0 then the normal form of w is accepted by Δ_n.*

Proof. Let w be a word accepted by Δ_0. Then w factorises as w as $w = w_1 \cdots w_n$, where w_i is accepted by an X_k component of Δ_0. Thus $w_i \in \mathbb{F}(X_{k_i} \cup Z)$, with $k_i \neq k_{i+1}$. If $n = 0$, then $w = 1$ and the lemma holds. Suppose the lemma holds for all w such that $n \leq k$, where $k \geq 0$ and now suppose $n = k + 1$. Let v_i be the normal form of w_i. If $v_i \in \mathbb{F}(Z)$, for some i then there is a factorisation of w with at most k factors, and the result follows. Otherwise, from Lemma lem:loopstop it:loopstop2 5.4.2, Δ_n accepts $v_1 \cdots v_n$, which is the normal form of w. □

Proof of Theorem thm:membership *1.1.* First we prove statement it:membership 1. Construct the flower automaton $\mathcal{F}(K)$ of K and then its Stallings automaton Γ_K, as at the beginning of this Section. Let $\Delta_0 = \Gamma_K$ and input this to the loop to obtain output Δ_n. Given a word $w \in \mathbb{F}(X_1 \cup X_2 \cup Z)$ write w in normal form using Algorithm I. If $w \in K$ then there is a word v in the generators of K with $\pi(v) = \pi(w)$, so v is accepted by Γ_K. From Lemma lem:nfacc 5.5, the word w is accepted by Δ_n. From Lemma lem:tocfold it:fold2 5.3.2, $L(\Delta_n) = L(\Gamma_K)$, so if $w \notin K$, then $w \notin L(\Delta_n)$.

Statements it:uni-membership 2 and it:uni-membership 3 follow from the observation that we may first run Algorithm I, then the Loop. □

6 Computational complexity

sec:TC

The usual approach to the study of computational problems is to begin with Turing machines and decision problems, that is, computational problems whose answer is "yes" or "no". Membership problems are usually considered as particular cases of decision problems. Moreover, every decision problem can be turned into a membership problem, but this transformation may add complications and even change the nature of the problem. For example representing graphs by words over an alphabet is possible but inconvenient and sometimes misleading. This leads us to take the following, more general, view of computational problems.

Definition 6.1. *A* decision problem *is a pair* $\mathrm{Pr} = (I, N)$*, where* N *is a countable set of* inputs *for* Pr *and* $I \subset N$ *is the* positive part *of* Pr*. The* answer *to the problem* Pr *on input* $w \in N$ *is "yes", if* $w \in I$*, and "no" otherwise. A* search problem *is a pair* $\mathrm{Pr} = (R, N \times J)$*, where* N *and* J *are countable sets, and* $R \subset N \times J$ *is a binary predicate. Given an element* $w \in N$*, a* solution *to the search problem* Pr *is an element* $v \in J$ *such that* $(w, v) \in R$*; that is, such that* $R(w, v)$ *is true.*

For instance, in this paper we are interested in the decision and search versions of the membership problem for a subgroup of a group, as formulated in the Introduction, for the particular case of amalgams of free groups. Given a fixed subgroup K of G and a word $w \in G$ written in double coset normal form we want to decide whether or not w represents an element of the subgroup K. Here double coset normal forms of elements of G form a (countable) set N of inputs, and K forms the positive part of the problem. In the corresponging membership search problem, if K is generated by $Y =< k_1, \ldots, k_l >$ then, on input word w representing an element of K, a solution is an expression for w in terms of $k_1, \ldots, k_l$. Thus, if G is generated by X we may formulate the problem as $\mathrm{Pr}(R, N \times J)$, where N is the set of elements $u \in \mathbb{F}(X)$ such that u is in double coset normal form and represents an element of K, $R \subseteq \mathbb{F}(X) \times \mathbb{F}(Y)$ is the set $R = \{(u, v) : u =_G v\}$ and $J = \mathbb{F}(Y)$.

When we speak of a problem Pr, we mean either a decision problem or a search problem. We say Pr is *solved* by an algorithm A if, for all inputs of Pr the algorithm A outputs the answer to Pr or a solution for Pr, as appropriate. The problem Pr is said to be *solvable* or *decidable* if it is solved by some algorithm.

In general to study the complexity of an algorithm A, one compares the resources spent by A on input w to the size of w. In our case the resource is time, or more precisely, the number of steps required for A to deal with

w. The *time consumed by an algorithm* A on an input w is $T_A(w)$, the number of steps performed by an algorithm A on input w. To facilitate the measurement of inputs we define a *size function*, for a set I, to be a map $\sigma : I \to \mathbb{N}$, the nonnegative integers, such that the preimage of each integer is finite. The choice of the size function depends of course on the problem at hand. For instance, if the input w is a word in a free group, a natural choice for input size is the length of that word. On the other hand if the input is an automaton (or graph), the input size can be chosen as the number of vertices and/or edges.

Definition 6.2. *Let A be an algorithm, T_A its time function, I the set of inputs, and σ a size function for I. The* complexity of A *with respect to σ is the function $\mathcal{C}o_A : \mathbb{N} \to \mathbb{N}$ defined by $\mathcal{C}o_A(n) = \max_{\sigma(w)\le n} T_A(w)$.*

We are not usually interested in the precise complexity of an algorithm but rather in estimating its rate of growth. We say that a problem Pr has complexity $f(n)$ if it is solved by an algorithm A for which $\mathcal{C}o_A(n)$ is $O(f(n))$.

In the remainder of this secion we define some functions which are useful in the analysis of our algorithms. We write $\log(x)$ for the logarithm to the base 2 and make the following definition, where $log^m(x)$ denotes the m-fold composition of log with itself.

def:log **Definition 6.3.** *The function $log^* : \mathbb{N} \to \mathbb{N}$ is given by $\log^*(n) = k$, where $k = \min\{m \in \mathbb{N} : log^m(n) \le 1\}$.*

Note that $\log^*(2^n) = \log^*(n) + 1$, so $\log^*$ grows very slowly with n: so slowly in fact that for most practical purposes it can be considered a constant. We shall use the following result from [19].

Theorem 6.4 ([19, Theorem 1.6]). *Let Δ be a connected, directed, labelled graph, with V vertices and E edges. Then there is an algorithm that will produce a Stallings folding of Δ in time $O(E + (V + E)\log^*(V))$.*

We shall apply this theorem repeatedly below, in cases where we have an estimate for the number of edges of a graph. For a connected graph Δ, we have $V \le E + 1$, so that $E + (V + E)\log^*(V) \le E + (2E + 1)\log^*(E + 1)$ and $\log^*(E + 1) \le log^*(2^E) \le \log^*(E) + 1$. Hence, for large enough E $E + (V + E)\log^*(V) \le E + (2E + 1)(\log^*(E) + 1) \le 7E\log^*(E)$. Therefore, we may construct a Stallings folding for Δ in time $O(E\log^*(E))$. This leads us to define

$$t_1(n) = O(n \cdot log^*(n)). \tag{6.1}$$ t1

Let H be a finitely generated subgroup of $F = \mathbb{F}(X)$ and $Y = \{h_1, \ldots, h_m\}$ be a generating set for H, where $h_i \in F$. Consider the Stallings automaton $\Gamma = \Gamma_Y(H)$ for H with vertices $V\Gamma$, edge set $E\Gamma$ and the root vertex 1. Denote by N the total length of Y, i.e. $N = |h_1|+|h_2|+\ldots+|h_m|$. Notice that by construction of Γ we have $|V\Gamma| \leq N$, $|E\Gamma| \leq N$, and $|E\Gamma| \geq |V\Gamma|$. Given a set of generators Y for H, one can construct the Stallings automaton Γ for H in time $t_1(N)$, as above.

In what follows we will frequently use the time complexity of the construction of a spanning subtree of a graph, and also of taking graph products. Observe, that one can find a spanning forest of a graph Γ in time at most $t_2(|E\Gamma|)$, for an arbitrary graph Γ, or $t_2'(|V\Gamma|)$, if Γ is connected, where

$$t_2(n) = O(n \cdot log(n)) \text{ and } t_2'(n) = O(n^2), \tag{6.2}$$ t2

with a help of Kruskal's algorithm, or the Dijkstra-Jarník-Prim algorithm, respectively.

Computation of $\Gamma_1 \times \Gamma_2$ takes time at most $t_3(|V\Gamma_1|, |V\Gamma_2|)$, where

$$t_3(n_1, n_2) = O(n_1 \cdot n_2). \tag{6.3}$$ t3

We will repeatedly use functions t_1, t_2, t_2', and t_3 in what follows.

If $f(x) = O(g(x))$ then we write $O(f(x)) \preceq O(g(x))$. With this notation, if $f(x) = O(g(x))$ and $O(g(x)) \preceq O(h(x))$ then $f(x) = O(h(x))$. We use this notation (repeatedly) in the situation where n_1 and n_2 depend on x, with $n_1(x) \leq n_2(x)$, for all x, and f is a non-decreasing function. In this case $O(f \circ n_1(x)) \preceq O(f \circ n_2(x))$, and we abuse notation by writing $O(f(n_1)) \preceq O(f(n_2))$.

6.1 The time complexity of construction of double coset normal forms

sub:doubleCo_nf

Recall that F_1 and F_2 are free groups, with subgroups $H_1 \leq F_1$, $H_2 \leq F_2$ which have finite bases $Y_1 = \{h_1, \ldots, h_m\}$ and $Y_2 = \{h_1', \ldots, h_m'\}$, respectively. $Z = \{z_1, \ldots, z_m\}$ is a set disjoint from $F_1 \cup F_2$ and there are maps ϕ_k such that $\phi_1(z_i) = h_i$ and $\phi_2(z_i) = h_i'$, $i = 1, \ldots, m$. $G = F_1 \underset{H_1=H_2}{*} F_2$ is the free product of F_1 and F_2 amalgamating H_1 and H_2.

To distinguish the length of elements in alphabets X_k or Z, we write $|\cdot|_k$ or $|\cdot|_Z$, accordingly. Define N_1 and N_2 be the *lengths* of Y_1 and Y_2, that is $N_1 = |h_1|_1 + |h_2|_1 + \ldots + |h_m|_1$ and $N_2 = |h_1|_2 + |h_2|_2 + \ldots + |h_m|_2$. Let A_k be the Stallings automaton for H_k and let T_k be a spanning tree for the associated graph Γ_{A_k}, as above. Denote by $E(\Gamma_{A_k})$ the edge set of Γ_{A_k}, and by $V\Gamma_{A_k}$ its vertex set. Recall that sets of words L_{Q_k} and L_{T_k} were defined in sub:foldings 2.2, $k = 1, 2$. Let S_k be the set of double coset representatives of $H_k \leq F_k$, determined by T_k, and $S_k = S_k^{(1)} \cup S_k^{(2)}$, where $S_k^{(i)}$ is the set of representatives of type i, as given in Definitions 2.2 and 2.4. Let P_k' be the set of non-diagonal vertices of $\Gamma_{A_k} \times \Gamma_{A_k}$, let P_k be the set of $\sim$ representatives of elements of P_k', and let C_k be the set of connecting elements for elements of P_k', as defined in Definition 2.4.

The following lemma estimates the complexity of construction of A_k and of rewriting words using double coset representatives.

em:dctransversal **Lemma 6.5.** *Let $w \in F_k \smallsetminus H_k$. Given a finite basis Y_k of H_k, one can*

- *construct the Stallings folding Γ_{A_k} for H_k in time $O(N_k^2)$,*
- *and, given Γ_{A_k}, L_{T_k}, the set P_0 of representatives of the equivalence relation $\sim$, and the corresponding set C of connecting elements, rewrite w in double coset normal form in time $O(|w|_k)$.*

Therefore, given Y_k and w the total time required to write w in dc-normal form is $O(N_k^4 + |w|_k)$.

Proof. To rewrite an element w in a double coset normal form, it is sufficient to compute Γ_{A_k}, the set L_{T_k}, the set P_0, and the corresponding set C of connecting elements (see preprocessing steps, section 4.1). Given these structures, one can then proceed to rewrite $w \in F_k$ in normal form, using Algorithm I.

Construction of Γ_{A_k} takes $t_1(N_k)$ steps, and then computation of its square $\Gamma_{A_k} \times \Gamma_{A_k}$ can be completed in $t_3(|V\Gamma_{A_k}|, |V\Gamma_{A_k}|) \preceq t_3(N_k, N_k)$. A spanning subtree T_k for Γ_{A_k} can be found in $t_2'(|V\Gamma_{A_k}|) \preceq t_2'(N_k)$ or $t_2(|ET_k|) \preceq t_2(|E\Gamma_{A_k}|) \preceq t_2(N_k)$; here the best estimate in terms of N_k is $O(N_k^2)$. Furthermore, the time required for computation of L_{T_k} is linear in $|ET_k|$. To construct the set P_0 of $\sim$ representatives (see definition 2.4), one must analyze all vertex sets of connected components of $\Gamma_{A_k} \times \Gamma_{A_k}$, and the time required for this procedure is at most $O(|V(\Gamma_{A_k} \times \Gamma_{A_k})|) \preceq O(N_k^2)$. To construct the set of connecting elements we construct a spanning forest for $\Gamma_{A_k} \times \Gamma_{A_k}$, which takes time $t_2'(N_k^2) = O(N_k^4)$, and once this has been done read off the connecting elements.

Given Γ_{A_k}, L_{T_k}, P_0 and the set C of connecting elements, the construction of which required time $O(N_k^4)$, we apply Algorithm I to $w \in F_k$. Analysis of Algorithm I shows that it takes time $O(|w|_k)$ to obtain the double coset normal form of w. Combining these estimates we obtain a total bound of $O(N_k^4 + |w|_k)$, as required. □

cor:dcnf_time **Corollary 6.6.** *Suppose $g \in G$ and $g = g_1 \cdots g_t$ is in reduced form. Given Y_1 and Y_2, one can rewrite g in double coset normal form in time $O(N_1^4 +$*

$N_2^4 + |g_1|_{\varepsilon_1} + |g_2|_{\varepsilon_2} + \ldots + |g_t|_{\varepsilon_t})$, *where* $\varepsilon_i \in \{1,2\}, i = 1,\ldots,t$, *and* $\varepsilon_i = k$ *if* $g_i \in F_k$.

Proof. Notice first, that if $g = g_1 \in H_k$, then we only rewrite it in terms of alphabet Z which takes time at most $O(|g_1|_k)$, and the case $g \in F_k \setminus H_k$ is covered by lemma 6.5, so suppose $g \notin F_k$, $k = 1,2$.

One can rewrite each syllable $g_i \in F_k$, of g, in normal form $g_i = h_{i,1} d_i h_{i,2}$, where $d_i \in S$ and $h_{i,j} \in H_1 \cup H_2$, in time $O(N_k^4 + |g_i|_k)$. Using ϕ_1^{-1} or ϕ_2^{-1}, we write $h_{i,1}$ and $h_{i,2}$ as reduced words in $\mathbb{F}(Z)$ (for $i = 1,\ldots,t$) and find the reduced word $h_{i,2}h_{(i+1),1} \in \mathbb{F}(Z)$ (for $i = 1,\ldots,t-1$) in time at most $O(|g_i|_{\varepsilon_i} + |g_i|_{\varepsilon_{i+1}})$. Thus the complexity of the process is $O(N_1^4 + N_2^4 + |g_1|_{\varepsilon_1} + |g_2|_{\varepsilon_2} + \ldots + |g_t|_{\varepsilon_t})$. □

6.2 The complexity of Algorithm II

sub:resolution

Let $K = \langle k_1,\ldots,k_s\rangle$, where k_i is an element of G written in normal form (3.1): $k_i = h_{i,0}t_{i,1}h_{i,1}\cdots t_{i,m_i}h_{i,m_i+1}$, with $h_{i,j} \in \mathbb{F}(Z)$ and $t_{i,j} \in S_1 \cup S_2$ as in section 3.1. Recall that Γ is the rooted graph, associated to the flower automaton $\mathcal{F}(K)$ of the subgroup $\hat{K} < \mathbb{F}(X_1 \cup X_2 \cup Z)$ generated by $k_1,\ldots,k_s$. Consider the Stallings folding Γ_K of Γ. Clearly, one can construct Γ_K in time $t_1(M)$, with $M = M_Z + M_1 + M_2$, where $M_Z = \sum_{i=1}^{s}\sum_{j=1}^{m_i} |h_{i,j}|_Z$, and M_k is the total length of $t_{i,j} \in S_k$ in terms of X_k for all appropriate i,j, i.e. $M_k = \sum_{i,j:t_{i,j}\in S_k} |t_{i,j}|_k$, with the function t_1 defined in (6.1); $k = 1,2$.

Let $\Delta_{(0)} = \Gamma_K$, and assume that the loop has been run n times to produce an inverse automaton $\Delta = \Delta_{(n)}$. We shall first study the complexity of modifications 1 — 5 for a subgraph Δ_k of Δ ($k = 1,2$), and then apply this result to estimate the general complexity of Algorithm II. For the sake of reader's convenience we refer to the steps C2 — C16 of the summary of Algorithm II, Section 4.2, when appropriate. Denote by $E_k\Delta$, $E_Z\Delta$ and $V\Delta$ the sets of edges of type X_k and of type Z and the set of vertices, respectively, of Δ. We use the notation $E = E_k = |E_k\Delta| + |E_Z\Delta_k|$, $e = |E_Z\Delta|$, $V = |V\Delta|$, and $q_k = |V\Gamma_{A_k}|$, $r_k = |X_k|$ for short.

lem:resolution

Lemma 6.7. *Suppose that* $\Delta = \Delta_{(n)}$, *that* Γ_{A_k}, T_k *and* P'_k, P_k, C_k, Δ_k, *and* ν *have been constructed. Then the time required to transform* Δ_k *into* $\Delta_{k,5}$ *is*

$$O((V + R_k e)^2 \cdot (2r_k)^{(V+R_k e)q_k+1}) +$$
$$+ O(m_1 V^5 + m_2 V^4 e + m_3 V^3 e^2 + m_4 V^2 e^3 + m_5 V e^4),$$

where $m_1,\ldots,m_5, q_k, r_k, R_k$ *are constants dependent on* Γ_{A_k} *and the basis* Y_k *for* H_k.

Proof. Let $R_k = \max\{|\phi_k(z_1)|_k, \ldots, |\phi_k(z_m)|_k\}$. Thus, as every petal of the Stallings automaton Γ_{A_k} has at most R_k edges, the maximal length of a path starting in the root vertex and terminating at arbitrary vertex of the tree T_k is at most $R_k - 1$.

1. $\Delta_k \rightsquigarrow \Delta_{k,1}$**.** (Steps C2–C3) At this stage we first add X_k-edges to Δ_k, to form a new graph $\Delta'_{k,1}$, and then fold $\Delta'_{k,1}$, to form $\Delta_{k,1}$. We add at most $R_k \cdot e$ edges of type X_k and at most $(R_k - 1) \cdot e$ new vertices to obtain $\Delta'_{k,1}$. It takes time at most $t_1(R_k \cdot E)$ to fold $\Delta'_{k,1}$ to $\Delta_{k,1}$. Clearly,

$$\begin{aligned} &|E\Delta_{k,1}| \leq |E\Delta'_{k,1}| \leq |E_k\Delta_k| + R_k \cdot e \leq R_k \cdot E \\ &\qquad\qquad \text{and} \\ &|V\Delta_{k,1}| \leq |V\Delta'_{k,1}| \leq V + R_k \cdot e \end{aligned} \tag{6.4}$$

evth1

Hence the time $t_{\Delta_{k,1}}$ required for the construction of $\Delta_{k,1}$ is

$$t_{\Delta_{k,1}} \preceq t_1(|E\Delta'_{k,1}|) + O(R_k \cdot E) = t_1(R_k \cdot E). \tag{6.5}$$

theta1

2. $\Delta_{k,1} \rightsquigarrow \Delta_{k,2}$**.** (Steps C4–C7) The time required for the construction of $\mathcal{P}_{k,1} = \Delta_{k,1} \times \Gamma_{A_k}$ is $t_3(|V\Delta_{k,1}|, q_k)$. Further, one can choose a spanning subforest $\Upsilon_{k,1}$ of $\mathcal{P}_{k,1}$ in $t_2(|V\mathcal{P}_{k,1}|)$ steps. Without loss of generality one can assume that $\mathcal{P}_{k,1}$ has precisely one connected component (which is the worst case from the point of view of the algorithm's complexity). Notice that $|V\mathcal{P}_{k,1}| \leq |V\Delta_{k,1}| \cdot q_k$.

Fix a pair (i,j) such that $i \leq j$ and consider the corresponding pair of vertices α_i, α_j of Δ. For this pair, we check whether or not $(\alpha_i, 1)$ and $(\alpha_j, 1)$ are connected by a simple path p. Since p is a simple (reduced) path in $\mathcal{P}_{k,1}$, it can be no longer than Q_k, where Q_k is the maximal (edge) length of a simple path in $\mathcal{P}_{k,1}$; so $Q_k \leq |V\mathcal{P}_{k,1}|$. Moreover, each such simple path p must have a label $l_p \in \mathbb{F}(X_k)$, accepted by $(\mathcal{P}_{k,1}, (\alpha_i, 1), (\alpha_j, 1))$. Hence, at this stage we must make at most B_{Q_k} checks, where B_n is the number of elements in H_k of length $\leq n$: so B_n is bounded above by the number of such elements in F_k. Thus $B_{Q_k} \leq 1 + \sum\limits_{i=0}^{Q_k} 2r_k \cdot (2r_k - 1)^i$. In the worst case we have $\frac{|V\Delta_{k,1}| \cdot (|V\Delta_{k,1}|+1)}{2}$ possible pairs (i,j) to analyze. Let $\overline{R}_k = \max\limits_{l_p \in B_{Q_k}} |\phi_k^{-1}(l_p)|_Z$. The time required for construction of $\Delta'_{k,2}$ is therefore $t_{\Delta'_{k,2}}$, where

$$\begin{aligned} t_{\Delta'_{k,2}} &= O(|V\Delta_{k,1}| \cdot (|V\Delta_{k,1}|+1) \cdot B_{Q_k} \cdot \overline{R}_k) \preceq \\ &\preceq O(|V\Delta_{k,1}| \cdot (|V\Delta_{k,1}|+1) \cdot r_k \cdot (2r_k - 1)^{|V\Delta_{k,1}| \cdot q_k} \overline{R}_k) \end{aligned} \tag{6.6}$$

pretheta'2

Using (6.4), obtain

$$t_{\Delta'_{k,2}} \preceq O((V+R_k e)^2 \cdot (2r_k)^{(V+R_k e)\cdot q_k+1}) \tag{6.7}$$ theta'2

One can fold this graph in time $t_1(|E\Delta'_{k,2}|)$.

Now we estimate the numbers of edges and vertices of $\Delta'_{k,2}$ and $\Delta_{k,2}$. In forming $\Delta'_{k,2}$ we do not add any X_k edges and so

$$\begin{aligned} |E\Delta_{k,2}| \le |E\Delta'_{k,2}| \le |E\Delta_{k,1}| + \overline{R}_k \cdot B_{Q_k} \cdot \frac{|V\Delta_{k,1}|\cdot(|V\Delta_{k,1}|+1)}{2} \le \\ \le R_k \cdot E + \overline{R}_k \cdot B_{Q_k} \cdot (V^2+e^2+V\cdot e), \end{aligned} \tag{6.8}$$ etheta2

$$\begin{aligned} |V\Delta_{k,2}| \le |V\Delta'_{k,2}| \le |V\Delta_{k,1}| + (\overline{R}_k-1) \cdot B_{Q_k} \cdot \frac{|V\Delta_{k,1}|\cdot(|V\Delta_{k,1}|+1)}{2} \le \\ \le V + R_k \cdot e + \overline{R}_k \cdot B_{Q_k} \cdot (V^2+e^2+V\cdot e), \end{aligned} \tag{6.9}$$ vtheta2

where the constants Q_k, B_{Q_k} and $\overline{R}_k$ are determined by $\Delta_{k,1}, H_k$, and F_k.

The total complexity of this stage of the modification process is therefore

$$t_{\Delta_{k,2}} = t_3(V+R_k\cdot e, q_k) + t_2((V+R_k\cdot e)\cdot q_k) + t_1(|E\Delta'_{k,2}|) + t_{\Delta'_{k,2}}, \tag{6.10}$$ theta2

where $|E\Delta'_{k,2}|$ is given by (6.8) and $t_{\Delta'_{k,2}}$ is estimated in formula (6.7).

3. $\Delta_{k,2} \rightsquigarrow \Delta_{k,3}$**.** (Steps C8–C10) We construct both $\mathcal{P}_{k,2} = \Delta_{k,2} \times \Gamma_{A_k}$ and a spanning subforest $\Upsilon_{k,2}$ of $\mathcal{P}_{k,2}$, using previously obtained $\mathcal{P}_{k,1}$ and $\Upsilon_{k,1}$. Suppose again $\mathcal{P}_{k,2}$ has only one non-trivial connected component. Let $V'\mathcal{P}_{k,2}$ denote the set of vertices of $\mathcal{P}_{k,2}$ of valency at least one (that is the set of non-isolated vertices). Since $|V\mathcal{P}_{k,2}| = |V\Delta_{k,2}|\cdot q_k$, then $|V'\mathcal{P}_{k,2}| \le |V\Delta_{k,2}|\cdot q_k$. As Γ_{A_k} is folded, so is $\mathcal{P}_{k,2}$, so vertices of $\mathcal{P}_{k,2}$ have degree at most $2r_k$. Hence the number of edges in $\mathcal{P}_{k,2}$ can be bounded above, in terms of the numbers of vertices of set of $\Delta_{k,2}$ and Γ_{A_k}, as follows.

$$|E\mathcal{P}_{k,2}| \le |V\mathcal{P}_{k,2}|\cdot r_k = |V\Delta_{k,2}|\cdot q_k \cdot r_k.$$

Then

$$|E\mathcal{P}_{k,2} \smallsetminus \Upsilon_{k,2}| \le |V\Delta_{k,2}|\cdot q_k \cdot r_k$$

while $|E\Upsilon_{k,2}| = |V'\mathcal{P}_{k,2}| - 1 < |V\Delta_{k,2}|\cdot q_k$. Every reduced path in $\Upsilon_{k,2}$ is simple, thus no longer than $|E\Upsilon_{k,2}|$ (in terms of X_k).

Fix a vertex $\alpha \in V\Delta_k$. For this vertex we consider edges $e = ((\alpha_i,\beta_1), x, (\alpha_j,\beta_2))$ of $E\mathcal{P}_{k,2} \smallsetminus E\Upsilon_{k,2}$, such that there is a path in $\mathcal{P}_{k,2}$, based at the vertex $(\alpha,1)$, with label $h_e = l_i x l_j$, as described in the definition of $\Delta_{k,3}$. Every such path has length at most $2|E\Upsilon_{k,2}|+1$. Let

$\overline{Q}_k = \max\limits_{h_e \in B_{2|E\Upsilon_{k,2}|+1}} |\phi_k^{-1}(h_e)|_Z$. Then, for all $\alpha \in V\Delta_k$ we must check at most $|E\mathcal{P}_{k,2}|$ edges and add a path of length at most $\overline{Q}_k$; which may be done in time $O(\overline{Q}_k + 2|E\Upsilon_{k,2}| + 1)$. Thus the time $t_{\Delta'_{k,3}}$ required for the construction of $\Delta'_{k,3}$ is

$$\begin{aligned} t_{\Delta'_{k,3}} &= O(|V\Delta_k| \cdot |E\mathcal{P}_{k,2} \smallsetminus \Upsilon_{k,2}| \cdot (\overline{Q}_k + 2|E\Upsilon_{k,2}| + 1)) \preceq \\ &\preceq O(V \cdot (|V\Delta_{k,2}| \cdot q_k \cdot r_k) \cdot (\overline{Q}_k + |V\Delta_{k,2}| \cdot q_k)) \preceq \\ &\preceq O(m_1V^5 + m_2V^4e + m_3V^3e^2 + m_4V^2e^3 + m_5Ve^4). \end{aligned} \tag{6.11}$$

theta'3

Here $m_1, \dots, m_5$ are polynomials on $q_k, r_k, \overline{Q}_k, \overline{R}_k, R_k$ and B_{Q_k}. One can fold this graph in time $t_1(|E\Delta'_{k,3}|)$.

For each vertex of $V\Delta_k$ we add at most $|E\mathcal{P}_{k,2}|$ paths, each of length at most $\overline{Q}_k$, so the number of edges and vertices of $\Delta'_{k,3}$ and $\Delta_{k,3}$ are

$$\begin{aligned} |E\Delta_{k,3}| \leq |E\Delta'_{k,3}| &\leq |E\Delta_{k,2}| + V \cdot \overline{Q}_k \cdot |V\Delta_{k,2}| q_k r_k \leq \\ &\leq R_k(\overline{Q}_k V e r_k q_k + E) + \overline{Q}_k V^2 r_k q_k + \\ &+ (V^2 + e^2 + Ve) \cdot (\overline{R}_k B_{Q_k}(\overline{Q}_k V r_k q_k + 1)), \end{aligned} \tag{6.12}$$

etheta3

and

$$\begin{aligned} |V\Delta_{k,3}| \leq |V\Delta'_{k,3}| &\leq |V\Delta_{k,2}| + V \cdot (\overline{Q}_k - 1) \cdot |V\Delta_{k,2}| \cdot rq \leq \\ &\leq V + \overline{Q}_k V^2 rq + \\ &+ (\overline{Q}_k Vrq + 1) \cdot (R_k e + \overline{R}_k B_{Q_k} \cdot (V^2 + e^2 + Ve)), \end{aligned} \tag{6.13}$$

vtheta3

where the constant $\overline{Q}_k$ is determined by $\Delta_{k,2}, H_k$, and F_k.

The total complexity of this stage of the modification process is

$$t_{\Delta_{k,3}} = t_1(|E\Delta'_{k,3}|) + t_{\Delta'_{k,3}}, \tag{6.14}$$

theta3

where $|E\Delta'_{k,3}|$, $t_{\Delta'_{k,3}}$ are given by (6.12) and (6.11), respectively.

4. $\Delta_{k,3} \rightsquigarrow \Delta_{k,4}$**.** (Steps C8–C10) We construct $\mathcal{P}_{k,3} = \Delta_{k,3} \times \Gamma_{A_k}$ and a spanning subforest $\Upsilon_{k,3}$ of $\mathcal{P}_{k,3}$. Since $E\Upsilon_{k,3} = E\Upsilon_{k,2}$ and $V\mathcal{P}_{k,3}$ coincides with $V\mathcal{P}_{k,2}$ plus some new isolated vertices, we use both $\mathcal{P}_{k,2}$ and $\Upsilon_{k,2}$ obtained above. Observe that by construction $E\Upsilon_{k,3} = E\Upsilon_{k,2}$.

As in Modification 4, fix a pair of vertices $\gamma = (\alpha_i, 1)$ and $\delta = (\alpha_j, \beta)$ in $\mathcal{P}_{k,3}$. Note that the number of such pairs is at most $V^2 \cdot (q_k - 1)$. In the worst case the pair γ, δ is not covered by $\mathcal{P}_{k,3}$, and, in the notation of Modification 4, we find $h = ab^{-1}$, where a is the label of a simple path from γ to δ in $\Upsilon_{k,3}$ and b is the label of a simple path in T_k. Thus $|h|_k = |ab^{-1}|_k \leq R_k + |E\Upsilon_{k,2}|$.

Then we add, to $\Delta_{k,3}$, a path with label wb, where the length of w is at most $\overline{Q}'_k$, for $\overline{Q}'_k = \max\limits_{h \in B_{R_k + |E\Upsilon_{k,2}|}} |\phi_k^{-1}(h)|_Z$. The time $t_{\Delta'_{k,4}}$ required to construct $\Delta'_{k,4}$ is the time it takes to perform this process as γ, δ ranges over all possible pairs of vertices. That is

$$t_{\Delta'_{k,4}} = O(V^2 q_k \cdot (\overline{Q}'_k + R_k)). \tag{6.15}$$

theta'4

This new graph can be folded in time $t_1(|E\Delta'_{k,4}|)$. Here the numbers of vertices and edges of $\Delta'_{k,4}$ and $\Delta_{k,4}$ are estimated by

$$\begin{aligned} |E\Delta_{k,4}| \leq |E\Delta'_{k,4}| &\leq |E\Delta_{k,3}| + V^2 \cdot (\overline{Q}'_k + R_k) \cdot (q_k - 1) < \\ &< R_k(\overline{Q}_k V e r_k q_k + E) + q_k V^2 \cdot (R_k + \overline{Q}'_k + \overline{Q}_k r_k) + \\ &+ \overline{R}_k B_{Q_k} \cdot (V^2 + e^2 + Ve) \cdot (\overline{Q}_k V r_k q_k + 1), \end{aligned} \tag{6.16}$$

etheta4

and

$$\begin{aligned} |V\Delta_{k,4}| \leq |V\Delta'_{k,4}| &\leq |V\Delta_{k,3}| + (\overline{Q}'_k + R_k - 1) \cdot V^2 \cdot (q_k - 1) < \\ &< V + (\overline{Q}_k V r q + 1) \cdot (R_k e + \overline{R}_k B_{Q_k} \cdot (V^2 + e^2 + Ve)) + \\ &+ V^2 q_k (\overline{Q}_k r_k + \overline{Q}_k + R_k), \end{aligned} \tag{6.17}$$

vtheta4

where the constant $\overline{Q}'_k$ is determined by $\Upsilon_{k,2}$, H_k, and F_k. Therefore,

$$t_{\Delta_{k,4}} = t_1(|E\Delta'_{k,4}|) + t_{\Delta'_{k,4}}, \tag{6.18}$$

theta4

where $t_{\Delta'_{k,4}}$ is given by (6.15).

5. $\Delta_{k,4} \rightsquigarrow \Delta_{k,5}$**.** (Steps C8–C10) The time required to construct $\mathcal{P}_{k,4} = \Delta_{k,4} \times \Gamma_{A_k}$ is $t_3(|V\Delta_{k,4}|, q_k)$. Further, a spanning subforest $\Upsilon_{k,4}$ of $\mathcal{P}_{k,4}$ can be constructed in time $t_2(|V\mathcal{P}_{k,4}|)$. We are assuming that a spanning tree T_k of Γ_{A_k}, the set of non-diagonal elements P'_k of $\Gamma_{A_k} \times \Gamma_{A_k}$, the set P_k of $\sim$ representatives elements of P'_k, and the set of connecting elements C_k have been constructed. Note that the number of elements of P'_k is $q_k^2 - q_k$.

As in Modification 5, fix an element $(\varepsilon, \xi) \in P'_k$. Let $c = c(\varepsilon, \xi)$ and assume that a_1, a_2, b_1 and b_2 are chosen as in Modification 5. Then $|c|_k \leq 2R_k$, while $|a_1|_k, |a_2|_k, |b_1|_k, |b_2|_k \leq R_k$. Thus, $|h_i|_k = |b_i c a_i^{-1}|_k < 4R_k$ for $i = 1, 2$. Let $\overline{Q}''_k = \max\limits_{h \in B_{4R_k - 1}} |\phi_k^{-1}(h)|_Z$. For every pair (ε, ξ) we add to $\Delta_{k,4}$ a path q of bounded length $|l_q| = |w_1 a_1 a_2^{-1} w_2^{-1}| \leq 2\overline{Q}''_k + 2R_k$. The time $t_{\Delta'_{k,5}}$ required to perform these transformations for all elements of P'_k, and so construct $\Delta'_{k,5}$, is therefore

$$\begin{aligned} t_{\Delta'_{k,5}} &= O(q_k^2 - q_k) \cdot (2\overline{Q}''_k + 2R_k) \\ &\preceq O(q_k^2 \cdot (\overline{Q}''_k + R_k)). \end{aligned} \tag{6.19}$$

theta'5

The graph $\Delta'_{k,5}$ can be folded in time $t_1(|E\Delta'_{k,5}|)$.

The number of vertices and edges of $\Delta'_{k,5}$ and $\Delta_{k,5}$ are estimated as

$$|E\Delta_{k,5}| \le |E\Delta'_{k,5}| \le |E\Delta_{k,4}| + 2(\overline{Q}''_k + R_k)\cdot(q_k^2 - q_k), \tag{6.20}$$ etheta5

and

$$|V\Delta_{k,5}| \le |V\Delta'_{k,5}| \le |V\Delta_{k,4}| + 2(\overline{Q}''_k + R_k)\cdot(q_k^2 - q_k), \tag{6.21}$$ vtheta5

where the constant $\overline{Q}''_k$ is determined by H_k, and F_k and $|E\Delta_{k,4}|$, $|V\Delta_{k,4}|$ are estimated in (6.16), (6.17).

Therefore,

$$t_{\Delta_{k,5}} = t_3(|V\Delta_{k,4}|, q_k) + t_2(|V\mathcal{P}_{k,4}|) + t_1(|E\Delta'_{k,5}|) + t_{\Delta'_{k,5}}, \tag{6.22}$$ theta5

where $t_{\Delta'_{k,5}}$ is given by (6.19) and $|V\mathcal{P}_{k,4}| \le |V\Delta_{k,4}|\cdot q_k$.

The total complexity of modifications of Δ_k. The bounds on the number of edges and vertices of the graphs constructed in Modifications 1–5 are non-decreasing, so the total complexity t_{Δ_k} of all modifications of Δ_k can be estimated as

$$\begin{aligned} t_{\Delta_k} &\preceq t_3(|V\Delta_{k,4}|, q_k) + t_2(|V\Delta_{k,4}|q_k) + t_1(|E\Delta'_{k,5}|) + t_{\Delta_{k,1}} + t_{\Delta'_{k,2}} + t_{\Delta'_{k,3}} + t_{\Delta'_{k,4}} + t_{\Delta'_{k,5}} \\ &\preceq O(|V\Delta_{k,4}|\cdot log(q_k|V\Delta_{k,4}|)) + O(|E\Delta'_{k,5}|\cdot log^*(|E\Delta'_{k,5}|)) + \\ &+ O((V+R_ke)^2\cdot(2r_k)^{(V+R_ke)q_k+1}) + \\ &+ O(m_1V^5 + m_2V^4e + m_3V^3e^2 + m_4V^2e^3 + m_5Ve^4) \preceq \\ &\preceq O((V+R_ke)^2\cdot(2r_k)^{(V+R_ke)q_k+1}) + \\ &+ O(m_1V^5 + m_2V^4e + m_3V^3e^2 + m_4V^2e^3 + m_5Ve^4), \end{aligned} \tag{6.23}$$ thetafin

where $m_1, \ldots, m_5$ are polynomials on $q_k, r_k, \overline{Q}_k, \overline{R}_k, R_k$ and B_{Q_k}, and $|V\Delta_{k,4}|$, $|E\Delta'_{k,5}|$ are given by (6.17) and (6.20), respectively.

□

Reassembly: construction of Δ'_5. (Step C17.) The time required to construct $\Delta_{1,5}\cup\Delta_{2,5}$ is bounded above by $t_{\Delta_1} + t_{\Delta_2}$. For each vertex (v,k) of Δ_k the set ν-im(v,k) has size at most V. Therefore the time required to construct Δ'_5 from $\Delta_{1,5}$ and $\Delta_{2,5}$ is at most

$$V(|V\Delta_{1,5}| + |V\Delta_{2,5}|)^2.$$

Hence the time required for Step C17 is at most

$$t_{\Delta'_5} = t_{\Delta_1} + t_{\Delta_2} + V(|V\Delta_{1,5}| + |V\Delta_{2,5}|)^2, \tag{6.24}$$ eq:D'_5

where bounds on t_{Δ_k} are given by (6.23), for $k = 1, 2$.
Reassembly: construction of Δ'_Z. (Step C18.) The number of vertices of Δ_Z is at most V, and as Δ'_5 has at most $|V\Delta_{1,5}| + |V\Delta_{1,5}|$ vertices the time taken to construct Δ'_Z from Δ_Z and Δ'_5 is at most

$$t_{\Delta'_Z} = V^2(|V\Delta_{1,5}| + |V\Delta_{2,5}|), \qquad (6.25)$$

eq:D'_Z

where, as before $|V\Delta_{1,5}|$ and $|V\Delta_{2,5}|$ can be estimated using (6.21), (6.20), (6.17) and (6.16).

Reassembly: construction of Δ''. (Step C19.) The time required to construct Δ'' from Δ'_5 and Δ'_Z is at most

$$t_{\Delta''} = |V\Delta'_5| \cdot |V\Delta'_Z| \le V(|V\Delta_{1,5}| + |V\Delta_{1,5}|). \qquad (6.26)$$

eq:D'

Total time for Algorithm II. The time t_Δ required to carry out Algorithm II, on input Δ, is thus bounded by

$$\begin{aligned} t_\Delta &= t_{\Delta'_5} + t_{\Delta'_Z} + t_{\Delta''} \\ &= t_{\Delta_1} + t_{\Delta_2} + V(|V\Delta_{1,5}| + |V\Delta_{2,5}|)^2 + V^2(|V\Delta_{1,5}| + |V\Delta_{2,5}|) + V(|V\Delta_{1,5}| + |V\Delta_{2,5}|)\end{aligned}$$

eq:alg2
eq:alg2

The numbers of vertices and edges of Δ'_5 can be estimated using (6.21), (6.20), (6.17) and (6.16) above. We have

$$\begin{aligned} |V\Delta'_5| &\le |V\Delta_{1,5}| + |V\Delta_{2,5}| \\ &\le \sum_{k=1}^{2}[|V\Delta_{k,4}| + 2(\overline{Q}''_k + R_k)\cdot(q_k^2 - q_k)], \text{ (from (6.21))} \\ &\le \sum_{k=1}^{2}[V + (\overline{Q}_k Vrq + 1)\cdot(R_k e + \overline{R}_k B_{Q_k}\cdot(V^2 + e^2 + Ve)) + \\ &\quad + V^2 q_k(\overline{Q}_k r_k + \overline{Q}_k + R_k)], \text{ (from (6.21))}.\end{aligned}$$

eq:alg2 eq:alg2

6.3 Complexity of the Loop

sub:loop

Each iteration of the loop involves input $\Delta = \Delta_{(n)}$, followed by folding of the output Δ'' of Step 1. The folding requires time $t_1(|E\Delta''|)$. Therefore the total time required to run the Loop once is $t_{\text{Loop}} = t_\Delta + t_1(|E\Delta''|)$.

ub:mp_complexity

6.4 Complexity of the membership problem

Now assume that $\Delta_{(0)} = \Gamma_K$ is input to the Loop. As Γ_K has at most M vertices, it has at most M components of type X_1 and X_2. The Loop is repeated only if $\Delta_{(n+1)}$ has fewer X_1 and X_2 components than $\Delta_{(n)}$: so there can be at most M iterations of the loop.

The number of vertices of $\Delta_{(n+1)}$ is at most $|V\Delta_{1,5}| + |V\Delta_{2,5}|$ and the number of edges of $\Delta_{(n+1)}$ is ...

Hence the nth iteration of the loop takes time ... .

Therefore the total time taken to produce Ψ is ...

References

BartholdiSilva [1] Laurent Bartholdi and Pedro V. Silva. Rational subsets of groups. *CoRR*, abs/1012.1532, 2010.

befe [2] M. Bestvina and M. Feighn. Bounding the complexity of simplicial group actions on trees. *Invent. Math.*, 103:449–469, 1991.

bez81 [3] V.N. Bezverkhnii. Solvability of the inclusion problem in a class of hnn-groups. *Algorithmic problems in the theory of groups and semigroups (Russian)*, pages 20–62, 1981.

bez86 [4] V.N. Bezverkhnii. Solution of the occurrence problem in some classes of groups with one defining relation. *Algorithmic problems in the theory of groups and semigroups (Russian)*, pages 3–21, 1986.

bez90 [5] V.N. Bezverkhnii. Solution of the occurrence problem in a certain class of groups i. *Problems in group theory and homological algebra (Russian)*, pages 40–53, 1990.

bez91 [6] V.N. Bezverkhnii. Solution of the occurrence problem in a certain class of groups ii. *Problems in group theory and homological algebra (Russian)*, pages 122–142, 1991.

BoWei [7] O. Bogopolski and R. Weidmann. On the uniqueness of factors of amalgamated products. *Journal of Group Theory*, 5(2):233–240, 2002.

KM02 [8] Ilya Kapovich and Alexei Myasnikov. Stallings foldings and subgroups of free groups. *IJAC*, 248:608–668, 2002.

KMW03 [9] Ilya Kapovich, Alexei Myasnikov, and Richard Weidmann. Foldings, graphs of groups and the membership problem. *IJAC*, 15(1):95–128, 2005.

Lawson04 [10] M.V. Lawson. *Finite automata*. Chapman & Hall/CRC, 2004.

LS [11] R.C. Lyndon and P.E. Schupp. *Combinatorial Group Theory*. Classics in Mathematics. Springer, 1977.

MKS [12] W. Magnus, A. Karrass, and D. Solitar. *Combinatorial Group Theory: Presentations Of Groups In Terms Of Generators And Relations*. Dover Books on Mathematics. Dover Publications, 2004.

mi58 [13] K. A. Mihailova. The occurrence problem for direct products of groups. *Dokl. Akad. Nauk SSSR*, 119:1103–1105, 1958.

mi59 [14] K. A. Mihailova. The occurrence problem for free products of groups. *Dokl. Akad. Nauk SSSR*, 127:746–748, 1959.

mi68 [15] K. A. Mihailova. The occurrence problem for free products of groups. *Matematicheskiy Sbornik*, 75 (117):199Ű–210, 1968.

SilvaWeil08 [16] P. Silva and P. Weil. On an algorithm to decide whether a free group is a free factor of another. *Theoretical Informatics and Applications*, 42:395–414, 2008.

stallings88 [17] J. R. Stallings. Foldings of g-trees. *Arboreal group theory*, pages 355–368, 1988.

stallings83 [18] J.R. Stallings. Toplogy of finite graphs. *Invent. Math.*, 71:551–565, 1983.

touikan06 [19] Nicholas W. M. Touikan. A fast algorithm for stallings' folding process. *IJAC*, 16(6):1031–1046, 2006.

ventura11 [20] E. Ventura. The lattice of subgroups of a free group, 2011. `https://www.epsem.upc.edu/~gagta5/pdfs/minic_Ventura.pdf`.

Andrew J. Duncan, School of Mathematics & Statistics, Newcastle University, Newcastle upon Tyne, NE1 7RU,UK
andrew.duncan@ncl.ac.uk
Elizaveta Frenkel, Moscow State University, GSP-1, Leninskie gory, 119991, Moscow, Russia
lizzy.frenkel@gmail.com