The Dehn function of Stallings' group

To appear in *Geometric and Functional Analysis*

Will Dison
Department of Mathematics,
The University of Bristol, University Walk, Bristol, BS8 1TW, United Kingdom
`w.dison@bristol.ac.uk`

Murray Elder
Mathematics,
The University of Queensland, Brisbane, Queensland 4072, Australia
`m.elder@uq.edu.au`

Timothy R. Riley*†
Department of Mathematics,
The University of Bristol, University Walk, Bristol, BS8 1TW, United Kingdom
`tim.riley@bris.ac.uk`

Robert Young
Institut des Hautes Études Scientifiques,
Le Bois Marie, 35 route de Chartres, F-91440 Bures-sur-Yvette, France
`rjyoung@ihes.fr`

Mathematics Subject Classification: 20F65
Keywords: Dehn function, Stallings' group, isoperimetric function, finiteness properties

Dedicated to John Stallings.

Abstract

We prove that the Dehn function of a group of Stallings that is finitely presented but not of type $\mathcal{F}_3$ is quadratic.

1 Introduction

A group is of type $\mathcal{F}_1$ when it can be finitely generated, $\mathcal{F}_2$ when it can be finitely presented, and more generally $\mathcal{F}_n$ when it admits an Eilenberg–Maclane space with finite n-skeleton. In the early 1960s Stallings [9] constructed a group S that is $\mathcal{F}_2$ but not $\mathcal{F}_3$. Bieri [2] recognised S to be

$$\mathrm{Ker}(\, F(\alpha,\beta) \times F(\gamma,\delta) \times F(\epsilon,\zeta) \ \twoheadrightarrow\ \mathbb{Z}\,) \tag{1}$$

---

[*] The third author is grateful for support from NSF grant DMS–0540830 and for the hospitality of the Institut des Hautes Études Scientifiques in Paris during the writing of this article.

[†] Corresponding author

where the map is that from the product of three rank-2 free groups to $\mathbb{Z} = \langle t \rangle$ which sends all six generators to t, and he showed that replacing $(F_2)^3$ by $(F_2)^n$ gives a family of groups (the *Bieri–Stallings* groups) of type $\mathcal{F}_{n-1}$ but not $\mathcal{F}_n$ [2].

Isoperimetric functions (defined below) for S have been established by a number of authors. Gersten proved that for $n \geq 3$, the Bieri–Stallings groups admit quintic isoperimetric functions [5]; this was sharpened to cubic by Baumslag, Bridson, Miller & Short in the case of S [1, §6]. Bridson [4] showed that the Bieri-Stallings groups were examples of a construction called *doubling* and argued that a class of doubles including these groups should also have quadratic isoperimetric functions. But Groves found an error in his proof [3, 6], and it seems that Bridson's approach, in fact, gives a cubic isoperimetric function, generalising the result in [1]. In this article we establish a quadratic isoperimetric function for S, and as S is not hyperbolic (as it is not of type $\mathcal{F}_3$, for example) this is best possible. And so we prove:

Theorem 1.1 *The Dehn function of Stallings' group is quadratic.*

More precisely, this theorem says that the Dehn function (defined below) of any finite presentation of Stallings' group is equivalent to $n \mapsto n^2$ in the following sense. For $f, g : \mathbb{N} \to \mathbb{N}$, we write $f \preceq g$ when $\exists C > 0, \forall n \in \mathbb{N}, f(n) \leq Cg(Cn + C) + Cn + C$, and we write $f \simeq g$ when $f \preceq g$ and $g \preceq f$. As is well-known, any two finite presentations of the same group have equivalent Dehn functions.

Our theorem fulfils Bridson's aim in [4] of exhibiting wild behaviour within the class of groups with quadratic Dehn functions —

Corollary 1.2 *There exists a group with quadratic Dehn functions that is not of type $\mathcal{F}_3$.*

Combined with results in [7, 8], the theorem also has the following corollaries.

Corollary 1.3 *The asymptotic cones of Stallings' group are all simply connected, but not all are 2-connected.*

Corollary 1.4 *Stallings' group admits a linear isodiametric function. Indeed, its filling length function is linear (that is, equivalent to $n \mapsto n$).*

We will work with the presentation

$$\langle\, a, b, c, d, s \;\mid\; [a, c],\ [a, d],\ [b, c],\ [b, d],\ s^a = s^b = s^c = s^d \,\rangle \tag{2}$$

for S of [1, 5]. Our notation is $[x, y] := x^{-1}y^{-1}xy$, $x^y := y^{-1}xy$, $x^{-y} := y^{-1}x^{-1}y$, and $s^a = s^b = s^c = s^d$ is shorthand for the six defining relations $s^a s^{-b}$, $s^a s^{-c}$, $s^a s^{-d}$, $s^b s^{-c}$, $s^b s^{-d}$, $s^c s^{-d}$. Note that these six relations can be rewritten as $[s, ab^{-1}], [s, ac^{-1}]$, and so on. One can view S as an HNN-extension of the product of free groups $F(a, b) \times F(c, d)$ with stable letter s commuting with all elements represented by words on $a^{\pm 1}, b^{\pm 1}, c^{\pm 1}, d^{\pm 1}$ of zero exponent-sum. The first four relations in the presentation are then the relations coming from $F(a, b) \times F(c, d)$, which we call *commutator relations*. [Gersten [5] showed that this is related to the description of S as a kernel (1) via $a = \epsilon\alpha^{-1}, b = \epsilon\beta^{-1}, c = \epsilon\gamma^{-1}, d = \epsilon\delta^{-1}, s = \zeta\epsilon^{-1}$.]

We prove Theorem 1.1 by presenting an algorithm (Algorithm 7) which takes as input a null-homotopic word of length n and transforms it to the empty word ε by applying relations from the presentation (2). We call the number of relations applied the *cost* and we wish to design the algorithm so that this is bounded by a constant multiple of n^2.

To understand the structure of the algorithm, it helps to understand the structure of a word which represents the identity. Since S is an HNN-extension of $F(a, b) \times F(c, d)$ by a generator s, by Britton's Lemma, a word w representing the identity contains "pinches", or pairs of letters s and s^{-1} separated by a word which commutes with s. (We will later call such words *balanced.*) Reducing w to the identity involves removing these pinches by bringing s's and s^{-1}'s together.

Using the presentation, one can show that s commutes with words of the form xy^{-1}, where $x, y \in \{a, b, c, d\}$; we will call a product of such words an *alternating word.* Then s can easily be commuted past a product of such words. The basic strategy of the algorithm is to identify a pinch, convert the balanced word inside to an alternating word, and cancel an s and s^{-1}. If there is a larger pinch containing the alternating word, we can repeat the process. Once we have removed all occurrences of the letters s and s^{-1} from w the resulting word will represent the identity in $F(a, b) \times F(c, d)$ and we will be able to apply commutator relations to convert this to the empty word. Provided that the process up to this point has not increased the length of the word significantly, the cost of this final step will be proportional to n^2.

The step which has the highest cost is converting a balanced word to an alternating word. One way to do this involves first separating a's and b's from c's and d's, then inserting a's and c's to produce an alternating word. For a word in a, b, c, d of length l, this has cost approximately l^2. If the pinches are deeply nested, we will need to repeat the process up to $n/2$ times, and as the second step largely undoes the first the total cost could be up to n^3.

We improve this by employing two key techniques. First, we utilise a divide-and-conquer strategy to convert balanced words to alternating form. We partition a balanced word into subwords and separate a's and b's from c's and d's in each subword, rather than in the whole word, before inserting a's and c's to make it alternating. We will say that the resulting word is in *partitioned alternating form.*

A typical intermediate stage in our process is a word with several subwords in partitioned alternating form. Indeed, we will specify intermediate stages by a list of subwords of the original word and partitions of these subwords; the intermediate stage will then be the original word with the specified subwords replaced by their partitioned alternating forms. As the algorithm progresses, these subwords grow and we merge adjacent subwords and adjacent pieces of the partitions. When two pieces in a partition of a balanced subword are merged, the cost is proportional to the square of the length of the words.

A priori, these merges could have a heavy total cost. To overcome this problem we employ a second key technique: we only use a particular type of partitioned alternating form, which we call *dyadic alternating form.* In this form, the partition of a balanced word only involves subwords of length 2^k, where $k \in \mathbb{N}$. Since once two pieces are merged together, they are never separated, there can be at most $n/2^k$ merges of pieces of length 2^{k-1}, and thus the total cost of all mergings will be proportional to

$$\sum_{k=1}^{\lceil \log n \rceil} \frac{n}{2^k}(2^{k-1})^2 \ \simeq \ n^2.$$

This article is organised as follows. In Section 2 we define alternating and balanced words, and we establish some basic facts about them. In Section 3 we define *dyadic alternating form* — this involves breaking up balanced words using a *dyadic partition*. Our main algorithm is in Section 5 and is analysed in Section 6. It proceeds by converting more and more of the input word w into dyadic alternating form by calling a number of subroutines (given in Section 4) to combine smaller subwords in dyadic alternating form into larger ones.

Article history. A number of prior versions of this article were made public. In the first, Elder and Riley established that $n^{5/2}$ is an isoperimetric function for S. Dison realised the result could be improved to $n^{7/3}$ and produced a new version of the paper in collaboration with Elder and Riley. Later Young contributed further insights that achieve the definitive n^2 result, and he, together with the other three authors, produced this version.

Acknowledgements. We are grateful to Noel Brady, Martin Bridson, Daniel Groves and Steve Pride for discussions on this problem, and to an anonymous referee for a careful reading.

2 Preliminaries

Write $u = u(a_1, \ldots, a_k)$ when u is a word on ${a_1}^{\pm 1}, \ldots, {a_k}^{\pm 1}$. Write $u[i]$ for the i-th letter of u. The length of u as a word (with no free reductions performed) is $\ell(u)$. The sum of the exponents of the letters in u is denoted by $\xi(u)$, which we call the *exponent sum* of u. Unless otherwise indicated, we consider two words to be equal when they are identical letter-by-letter. A *partition* of u is any way of expressing u as a concatenation $u_1 \ldots u_k$ of subwords. We denote the empty word by ε.

Definition 2.1 (Cost, Dehn function, isoperimetric function) *Given words w, w' representing the same element of a group with finite presentation $\langle \mathcal{A} \mid \mathcal{R} \rangle$, one can convert w to w' via a sequence of words $W = (w_i)_{i=0}^m$ in which $w_0 = w$, $w_m = w'$ and for each i, w_{i+1} is obtained from w_i by free reduction ($w_i = \alpha a a^{-1} \beta \mapsto \alpha\beta = w_{i+1}$ where $a \in \mathcal{A}^{\pm 1}$), by free expansion (the inverse of a free reduction), or by applying a relator ($w_i = \alpha u \beta \mapsto \alpha v \beta = w_{i+1}$ where a cyclic conjugate of uv^{-1} is in $\mathcal{R}^{\pm 1}$). The* cost *of W is the number of i such that $w_i \mapsto w_{i+1}$ is an* application-of-a-relator *move. For words w that represent the identity (i.e.*

null-homotopic *words*), Area(w) *is the minimal cost amongst all* W *converting* w *to* ε. *The* Dehn function $\delta : \mathbb{N} \to \mathbb{N}$ *of* $\langle \mathcal{A} \mid \mathcal{R} \rangle$ *is*

$$\delta(n) \ := \ \max\{\text{Area}(w) \mid w \text{ represents } 1 \text{ and } \ell(w) \leq n\}.$$

An isoperimetric function *for* $\langle \mathcal{A} \mid \mathcal{R} \rangle$ *is any* $f : \mathbb{N} \to \mathbb{N}$ *such that* $\delta(n) \leq f(n)$ *for all* n.

Definition 2.2 (Alternating words) *A word* $u = u(a, b, c, d)$ *is* alternating *if* u *has even length and* $u[i]$ *is in* $\{a, b, c, d\}$ *for all odd* i *and in* $\{a^{-1}, b^{-1}, c^{-1}, d^{-1}\}$ *for all even* i.

[The reader familiar with van Kampen diagrams and corridors (also known as bands) may find it helpful to note that alternating words are those which, after removing all $aa^{-1}, bb^{-1}, cc^{-1}$ and dd^{-1} subwords, can be read along the sides of s-corridors in van Kampen diagrams over S.]

Definition 2.3 (Balanced words) *A word* $u = u(a, b, c, d, s)$ *is* balanced *if there exists an alternating word* $v = v(a, b, c, d)$ *with* $u = v$ *in* S.

Lemma 2.4 *For a word* $u = u(a, b, c, d, s)$, *the following are equivalent.*

(*i*) *u is balanced.*

(*ii*) *u represents an element of* $\langle a, b, c, d \rangle$ *in* S *that commutes with* s.

(*iii*) *u represents an element of* $\langle a, b, c, d \rangle$ *in* S *and* $\xi(u) = 0$.

Proof: The equivalence of (*i*) and (*ii*) is straight-forward.

Alternating words have exponent-sum zero so (*i*) implies (*iii*) by the following observation. Every relation of presentation (2) has exponent-sum zero, so exponent-sum is preserved whenever a relation is applied to a word and hence if two words on $a^{\pm 1}, b^{\pm 1}, c^{\pm 1}, d^{\pm 1}, s^{\pm 1}$ represent the same element in S, then they have the same exponent sum.

To see that (*iii*) implies (*i*) we can convert a word $w = w(a, b, c, d)$ with $\xi(w) = 0$ to a word in alternating form as follows. Commute all a, b letters to the front to give a word $\rho(a, b)\sigma(c, d)$. For each letter of $\rho(a, b)$, replace a, b, a^{-1}, b^{-1} by $ac^{-1}, bc^{-1}, ca^{-1}, cb^{-1}$, respectively. For each letter of $\sigma(c, d)$, replace c, d, c^{-1}, d^{-1} by $ca^{-1}, da^{-1}, ac^{-1}, ad^{-1}$, respectively. In the middle insert $(ca^{-1})^{\xi(\rho(a,b))}$ which cancels out the c, a letters that were added. $\square$

Lemma 2.5 *Suppose a word* $w = w(a, b, c, d, s)$ *is expressed as* $w = \alpha v \beta$ *in which* v *is a balanced subword. Then* w *is balanced if and only if* $\alpha\beta$ *is balanced.*

Proof: Induct on the number of s letters in w. The base case where $w = w(a, b, c, d)$ is immediate and the induction step an application of Britton's Lemma. Alternatively, this result is an observation on the layout of s-corridors in a van Kampen diagram demonstrating that w equates to some alternating word in S. $\square$

Lemma 2.6 *If w is a balanced word of length at least 2, it contains a subword xy such that either $xy = s^{\pm 1}s^{\mp 1}$ or $x, y \in \{a, b, c, d\}^{\pm 1}$ with x and y having opposite exponents.*

Proof: By Britton's Lemma, w contains either a subword $s^{\pm 1}s^{\mp 1}$ or a non-empty balanced subword $u = u(a, b, c, d)$. In the second case, since u is balanced it has exponent-sum zero and so must contain a subword $u[i]u[i+1]$ where $u[i]$ and $u[i+1]$ have opposite exponent; take $xy = u[i]u[i + 1]$. □

3 Dyadic alternating form

The main algorithm will systematically convert subwords of the input word into a special alternating form, which we describe in this section.

By an *interval* in $\mathbb{Z}$, we mean $\mathbb{Z} \cap [\lambda, \mu]$ for some $\lambda, \mu \in \mathbb{R}$. For $r, j \in \mathbb{Z}$ with $r \geq 0$, define the *dyadic* interval $D_{r,j}$ to be $\mathbb{Z} \cap [j2^r, (j + 1)2^r)$, as illustrated in Figure 1. Note that $D_{r+1,j} = D_{r,2j} \cup D_{r,2j+1}$ and any two $D_{r,j}$ and $D_{r',j'}$ are either disjoint or one contains the other. The *height* of $D_{r,j}$ is r.

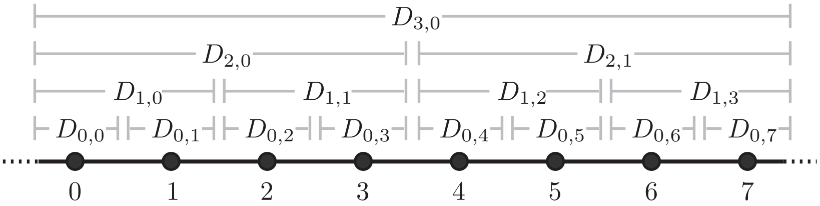

Figure 1: Dyadic partitions of the integers.

Given an interval $U \subset \mathbb{Z}$, a *cover* of U is a collection $\{U_i\}_{i=1}^n$ of disjoint intervals $U_i \subseteq U$ with $\cup_{i=1}^n U_i = U$. We say that the indexing on $\{U_i\}_{i=1}^n$ is *ascending* if $\min(U_{i+1}) = 1 + \max(U_i)$ for each i. Given a subword v of a word w, let $V \subset \mathbb{Z}$ be the interval consisting of the positions of the letters of v in w. Then there is an obvious 1-1 correspondence between partitions of v and covers of V.

A *dyadic cover* of an interval $U \subset \mathbb{Z}$ is a cover consisting of dyadic intervals. Define the *minimal dyadic cover* $\mathrm{MDC}(U)$ of U to be the set of maximal elements (with respect to containment) of $\{D_{r,j} | D_{r,j} \subseteq U\}$. It is clear that this set is a dyadic cover of U. In fact, as a consequence of Lemma 3.2, it is the dyadic cover with the minimal number of elements. As an example, if $U = \mathbb{Z} \cap [5, 20]$, then $\{D_{r,j} | D_{r,j} \subseteq U\}$ is

$$\{D_{0,5}, D_{0,6}, \ldots, D_{0,20}, D_{1,3}, D_{1,4}, \ldots, D_{1,9}, D_{2,2}, D_{2,3}, D_{2,4}, D_{3,1}\}$$

and

$$\begin{aligned}\mathrm{MDC}(U) &= \{D_{0,5}, D_{1,3}, D_{3,1}, D_{2,4}, D_{0,20}\} \\ &= \{5\} \cup \{6, 7\} \cup \{8, \ldots, 15\} \cup \{16, 17, 18, 19\} \cup \{20\}.\end{aligned}$$

In Figure 2 we display $\mathrm{MDC}(\mathbb{Z} \cap [5, 20])$ indicating the heights of its elements.

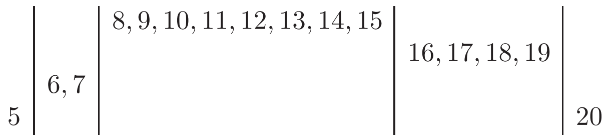

Figure 2: The minimal dyadic cover of $\mathbb{Z} \cap [5, 20]$.

Lemma 3.1 *Let* U *be an interval in* $\mathbb{Z}$*. Say* $\mathrm{MDC}(U) = \{U_1, \ldots, U_n\}$ *where the* U_i *are indexed in ascending order. Then for each* k *either*

$$\left|\bigcup_{i=1}^{k-1} U_i\right| < |U_k| \qquad \text{or} \qquad \left|\bigcup_{i=k+1}^{n} U_i\right| < |U_k|.$$

Proof: Let $r := \mathrm{height}(U_k)$. Assume k is not 1 or n, as otherwise the result is trivial. At most two consecutive U_i have the same height, so either U_{k-1} or U_{k+1} is at a different height to U_k. By the maximality of the elements of $\mathrm{MDC}(U)$, if one of U_{k-1} or U_{k+1} has height greater than r then the other must have height less than r. So $\mathrm{height}(U_{k-1}) < r$ or $\mathrm{height}(U_{k+1}) < r$.

As suggested by the example of Figure 2, $\mathrm{MDC}(U)$ resembles a pyramid, in that there is some m such that the sequence $\mathrm{height}(U_1), \ldots, \mathrm{height}(U_m)$ is strictly increasing and the sequence $\mathrm{height}(U_{m+1}), \ldots, \mathrm{height}(U_n)$ is strictly decreasing. So if $\mathrm{height}(U_{k-1}) < r$ then the sequence $\mathrm{height}(U_1), \ldots, \mathrm{height}(U_{k-1})$ is strictly increasing and

$$\left|\bigcup_{i=1}^{k-1} U_i\right| \leq \sum_{i=0}^{r-1} 2^i < 2^r = |U_k|.$$

The case where $\mathrm{height}(U_{k+1}) < r$ is similar. $\square$

In order to control the merging process mentioned in the introduction, we will use the following lemmas. The first is an observation on how dyadic covers can be converted to minimal dyadic covers.

Lemma 3.2 *Let* $U \subseteq \mathbb{Z}$ *be an interval and* X_0 *be a dyadic cover of* U*. Then there is a sequence* $X_0, \ldots, X_n$ *of dyadic covers of* U *with* $X_n = \mathrm{MDC}(U)$ *and, for all* i*,* X_{i+1} *is obtained from* X_i *by merging two adjacent dyadic intervals — that is, if* $X_i = \{S_1, \ldots, S_r\}$*, where the* S_j *are indexed in ascending order, then there exists* k *such that* $|S_k| = |S_{k+1}|$ *and* $X_{i+1} = \{S_1, \ldots, S_k \cup S_{k+1}, \ldots, S_r\}$*. In particular, if* $S \in X_i$*, then there exists* $W \in X_0$ *such that* $W \subseteq S$*.*

Proof: If $X_0 = \mathrm{MDC}(U)$, we are done. Otherwise, by the definition of $\mathrm{MDC}(U)$, there are dyadic intervals $W \in X_0$ and $D \subseteq U$ such that $W \subsetneq D$. Choose a minimal length such $W \in X_0$. The dyadic intervals containing W are well-ordered by containment; let D be the dyadic interval of minimal length strictly containing W. We then have $W \subsetneq D \subseteq U$ and $\ell(D) = 2\ell(W)$.

We claim that $W' := D \smallsetminus W$ is in X_0. We know that W' is a dyadic interval of the same length as W. Since $W' \subseteq U$ and X_0 covers U, there must be an interval $Z \in X_0$ such that $Z \cap W'$ is nonempty. We claim that $Z = W'$. Since Z and W' are dyadic and have nonempty intersection, one must contain the other. This containment, however, cannot be strict; on one hand, if $W' \subsetneq Z$, it must also contain W, so since X_0 is a collection of disjoint sets, Z cannot strictly contain W'. On the other hand, by the minimality of W, the set Z cannot strictly contain W'. Thus $W' = Z \in X_0$.

Then W and W' are adjacent intervals of equal length; without loss of generality, assume that W is to the left of W', so that X_0 can be expressed as

$$X_0 \;=\; \{S_1, \ldots, W, W', \ldots, S_r\}$$

Then we let

$$X_1 \;=\; \{S_1, \ldots, W \cup W', \ldots, S_r\}.$$

We repeat the process to construct $X_2, \ldots, X_n$. With each step, the number of elements in the partition decreases by one, and so the process terminates, and then every element of X_n is maximal among dyadic intervals contained in U, so $X_n = \mathrm{MDC}(U)$.

The last assertion in the lemma follows by induction on n: if $n = 0$, it is trivially true, and by construction, any element of X_{i+1} contains an element of X_i. □

We can now describe a process of merging two adjacent minimal dyadic covers. Note that all of the changes occur at the boundary between the two covers.

Corollary 3.3 *Let $U, V \subseteq \mathbb{Z}$ be adjacent intervals with $\min(V) = \max(U) + 1$. Then there exists a sequence $X_0, \ldots, X_n$ of dyadic covers of $U \cup V$ with*

$$X_0 \;=\; \mathrm{MDC}(U) \cup \mathrm{MDC}(V) \qquad \text{and} \qquad X_n \;=\; \mathrm{MDC}(U \cup V)$$

and, for each i, expressing X_i as $\{S_1, \ldots, S_r\}$ where the S_j are indexed in ascending order, there exists k such that $|S_k| = |S_{k+1}|$, $X_{i+1} = \{S_1, \ldots, S_k \cup S_{k+1}, \ldots, S_r\}$ and $S_k \cup S_{k+1}$ is not a subset of U or V.

Proof: $\mathrm{MDC}(U) \cup \mathrm{MDC}(V)$ is a dyadic cover, so we may apply Lemma 3.2 to obtain a sequence $X_0, \ldots, X_n = \mathrm{MDC}(U \cup V)$ of dyadic covers. It remains only to prove the final assertion. By contradiction, suppose that $S_k \cup S_{k+1} \subset U$. Then there is some $W \in \mathrm{MDC}(U) \cup \mathrm{MDC}(V)$ such that $W \subseteq S_k$. In fact, we must have $W \in \mathrm{MDC}(U)$. But this is impossible since $S_k \cup S_{k+1}$ is a dyadic interval strictly containing W and contained in U. Similarly, $S_k \cup S_{k+1}$ cannot be a subset of V. □

The following similar observation applies when one of the intervals consists of a single integer.

Corollary 3.4 *Suppose $U \subset \mathbb{Z}$ is an interval. Let $s = \min(U) - 1$ and $t = \max(U) + 1$. Then:*

1. *There exists a sequence* $X_0, \dots, X_n$ *of dyadic covers of* $\{s\} \cup U$ *with*

$$X_0 \;=\; \{\{s\}\} \cup \mathrm{MDC}(U) \quad \textit{and} \quad X_n \;=\; \mathrm{MDC}(\{s\} \cup U)$$

   and, for each i*, expressing* X_i *as* $\{S_1, \dots, S_r\}$ *where the* S_i *are indexed in ascending order, we find* $|S_1| = |S_2|$ *and* $X_{i+1} = \{S_1 \cup S_2, S_3, \dots, S_r\}$.

2. *There exists a sequence* $Y_0, \dots, Y_n$ *of dyadic covers of* $U \cup \{t\}$ *with*

$$Y_0 \;=\; \mathrm{MDC}(U) \cup \{\{t\}\} \quad \textit{and} \quad Y_n \;=\; \mathrm{MDC}(U \cup \{t\})$$

   and, for each i*, expressing* Y_i *as* $\{S_1, \dots, S_r\}$ *where the* S_i *are indexed in ascending order, we find* $|S_{r-1}| = |S_r|$ *and* $Y_{i+1} = \{S_1, \dots, S_{r-2}, S_{r-1} \cup S_r\}$.

Definition 3.5 (Partitioned Alternating Form) *Let* $v = v(a, b, c, d)$ *be a balanced word partitioned as* $v = v_1 \dots v_k$*. Let* λ_i *(resp.* μ_i*) be* v_i *with all letters* $c^{\pm 1}$ *and* $d^{\pm 1}$ *(resp.* $a^{\pm 1}$ *and* $b^{\pm 1}$*) deleted. Obtain* $\rho_i(a, b, c)$ *from* λ_i *by replacing each* a, b, a^{-1}, b^{-1} *by* $ac^{-1}, bc^{-1}, ca^{-1}, cb^{-1}$*, respectively. Obtain* $\sigma_i(a, c, d)$ *from* μ_i *by replacing each* c, d, c^{-1}, d^{-1} *by* $ca^{-1}, da^{-1}, ac^{-1}, ad^{-1}$*, respectively. The* partitioned alternating form *of* v *with respect to* $v_1 \dots v_k$ *is*

$$\tau \;:=\; \rho_1\,(ca^{-1})^{\xi(\lambda_1)}\sigma_1\,(ac^{-1})^{\xi(\lambda_1\mu_1)} \dots \rho_k\,(ca^{-1})^{\xi(\lambda_1\mu_1\dots\lambda_k)}\sigma_k.$$

Lemma 3.6 *Any partitioned alternating form for* v *equals* v *in* $F(a, b) \times F(c, d)$.

Proof: We will use the notation of Definition 3.5. Each $v_i = \lambda_i \mu_i$ in S and so $v = \lambda_1\mu_1 \dots \lambda_k\mu_k$ in $F(a, b) \times F(c, d)$.

Commutator relations can be used to convert a word $c^m\lambda\mu$, where $\lambda = \lambda(a, b)$, $\mu = \mu(c, d)$, and $m \in \mathbb{Z}$, to $\rho\,(ca^{-1})^{m+\xi(\lambda)}\,\sigma\,(ac^{-1})^{m+\xi\,(\lambda\mu)}\,c^{m+\xi(\lambda\mu)}$, where ρ and σ are obtained from λ and μ as per Definition 3.5. Making k successive such transformations, working from left to right and beginning with $m = 0$, converts $\lambda_1\mu_1 \dots \lambda_k\mu_k$ to τ. (Note that $\xi(\lambda_1\mu_1 \dots \lambda_k\mu_k) = 0$ as v is balanced.) ☐

Lemma 3.7 *In Definition* 3.5*, for all* i*, the prefix*

$$\rho_1\,(ca^{-1})^{\xi(\lambda_1)}\sigma_1\,(ac^{-1})^{\xi(\lambda_1\mu_1)} \dots \rho_i\,(ca^{-1})^{\xi(\lambda_1\mu_1\dots\lambda_i)}\,\sigma_i\,(ac^{-1})^{\xi(\lambda_1\mu_1\dots\lambda_i\mu_i)}$$

and suffix

$$\rho_{i+1}\,(ca^{-1})^{\xi(\lambda_1\mu_1\dots\lambda_{i+1})}\sigma_{i+1}\,(ac^{-1})^{\xi(\lambda_1\mu_1\dots\lambda_{i+1}\mu_{i+1})} \dots \rho_k\,(ca^{-1})^{\xi(\lambda_1\mu_1\dots\lambda_k)}\,\sigma_k$$

of τ *depend only on* $v_1 \dots v_i$ *and* $v_{i+1} \dots v_k$*, respectively.*

Proof: In the case of the prefix, this is self-evident. It is true for the suffix, because $\xi(\lambda_1\mu_1 \dots \lambda_k\mu_k) = 0$ as v is balanced, which allows one to express the exponents of (ca^{-1}) and (ac^{-1}) in terms of μ_{i+1}, λ_{i+2}, μ_{i+2}, $\dots$, λ_k, and μ_k. ☐

Definition 3.8 (Dyadic Alternating Form) *Let $v = v(a, b, c, d, s)$ be a balanced subword of a word w. Let $\hat{v}$ and $\hat{w}$ be v and w with all $s^{\pm 1}$ removed. Let V be the set of positions of the letters of $\hat{v}$ in $\hat{w}$. Define the* dyadic partition *of $\hat{v}$ to be the partition corresponding to the partition* $\mathrm{MDC}(V)$ *of V. The* dyadic alternating form *of v is defined to be the partitioned alternating form of $\hat{v}$ with respect to the dyadic partition.*

Note that the dyadic alternating form of v depends on its position in w.

Lemma 3.9 *The dyadic alternating form produced in the above definition equals v in S.*

Proof: Since v is balanced in this definition, $v = \hat{v}$ in S. The result then follows from Lemma 3.6. □

One might think that the $(ac^{-1})^{\xi(\rho_1 \cdots)}$ and $(ca^{-1})^{\xi(\rho_1 \cdots)}$ inserted could dramatically increase length, but the following estimates show this is not so for dyadic alternating form.

Lemma 3.10 *In the dyadic partition $v_1 \ldots v_k$ of $\hat{v}$ arising in Definition* 3.8*, for all i,*

$$|\xi(v_1 \ldots v_i)| < 2\ell(v_i) \quad \text{and} \quad |\xi(v_1 \ldots v_{i-1}\lambda_i)| < 2\ell(v_i).$$

Proof: As the v_i are defined using a minimal dyadic cover, Lemma 3.1 implies that either $\ell(v_1 \ldots v_{i-1}) < \ell(v_i)$ or $\ell(v_{i+1} \ldots v_k) < \ell(v_i)$.

In the first case, $|\xi(v_1 \ldots v_i)| \leq \ell(v_1 \ldots v_i) = \ell(v_1 \ldots v_{i-1}) + \ell(v_i) < 2\ell(v_i)$ and $|\xi(v_1 \ldots v_{i-1}\lambda_i)| \leq \ell(v_1 \ldots v_{i-1}\lambda_i) = \ell(v_1 \ldots v_{i-1}) + \ell(\lambda_i) < 2\ell(v_i)$.

In the second case, $\xi(v_{i+1} \ldots v_k) = -\xi(v_1 \ldots v_i)$ since $\xi(\hat{v}) = \xi(v) = 0$ as v is balanced. So $|\xi(v_1 \ldots v_i)| = |\xi(v_{i+1} \ldots v_k)| \leq \ell(v_{i+1} \ldots v_k) < \ell(v_i)$. Similarly, $\xi(\mu_i v_{i+1} \ldots v_k) = -\xi(v_1 \ldots v_{i-1}\lambda_i)$ and so $|\xi(v_1 \ldots v_{i-1}\lambda_i)| = |\xi(\mu_i v_{i+1} \ldots v_k) \leq \ell(\mu_i v_{i+1} \ldots v_k) < 2\ell(v_i)$. □

Lemma 3.11 *The dyadic alternating form of v defined in Definition* 3.8 *has length at most* $10\ell(v)$*.*

Proof: In the dyadic alternating form, each letter of $\hat{v}$ is matched with either an $a^{\pm 1}$ or $c^{\pm 1}$. As $\ell(\hat{v}) \leq \ell(v)$, this accounts for at most $2\ell(v)$ letters.

There are $2(|\xi(\lambda_1)| + |\xi(\lambda_1\mu_1)| + \ldots + |\xi(\lambda_1\mu_1 \ldots \lambda_k)|)$ further letters appearing in the powers of $(ac^{-1})^{\pm 1}$. But for all i,

$$\begin{aligned} & |\xi(\lambda_1\mu_1 \ldots \lambda_{i-1}\mu_{i-1}\lambda_i)| + |\xi(\lambda_1\mu_1 \ldots \lambda_i\mu_i)| \\ & \qquad = \ |\xi(v_1 \ldots v_{i-1}\lambda_i)| + |\xi(v_1 \ldots v_i)| \ < \ 4\ell(v_i) \end{aligned}$$

by Lemma 3.10. So it suffices to add

$$2(|\xi(\lambda_1)| + |\xi(\lambda_1\mu_1)| + \ldots + |\xi(\lambda_1\mu_1 \ldots \lambda_k)|) \ \leq \ \sum_{i=1}^{k} 8\ell(v_i) \ = \ 8\ell(\hat{v}) \ \leq \ 8\ell(v).$$

□

4 Subroutines

In this section we present the key subroutines that will be called by the main algorithm. Algorithms 1–6 manipulate words using the relations of $F(a,b) \times F(c,d)$, while Algorithm 7 uses the relators of the whole group S. Each algorithm gives a method for converting one word to another word by applications of relators, free expansions, and free reductions. Thus each algorithm has a cost in the sense of Definition 2.1.

The first algorithm gives a rough-and-ready scheme for converting words when one does not have any additional information about the structure of the input word. As such the cost of the algorithm is high.

Algorithm 1 Shuffling between words $F(a,b) \times F(c,d)$

Input: Words $u = u(a,b,c,d)$ and $v = v(a,b,c,d)$ representing the same elements of $F(a,b) \times F(c,d)$

Goal: Convert u to v.

Method: Shuffle the a's and b's in u to the front of the word and freely reduce to produce a word w. The same procedure would convert v to w, so run it in reverse to convert w to v.

Lemma 4.1 *The cost (in the sense of Definition 2.1) of the transformation of Algorithm 1 is at most* $\ell(u)^2 + \ell(v)^2$.

Proof: Each letter in u or v is shuffled past fewer than $\ell(u)$ or $\ell(v)$ other letters (respectively). ☐

Algorithm 2 Merging within partitioned alternating form

Input: A balanced word $v = v(a,b,c,d)$, a partition $v_1 \dots v_k$ of v, an integer $i \in \{1, \dots, k-1\}$, and the partitioned alternating form τ of v with respect to $v_1 \dots v_k$

Goal: Convert τ to the partitioned alternating form $\bar{\tau}$ of v with respect to the partition $v_1 \dots v_{i-1}\, \bar{v}\, v_{i+2} \dots v_k$ where $\bar{v} = v_i v_{i+1}$.

Method: We use the notation of Definition 3.5 and write

$$\tau \;=\; \rho_1\,(ca^{-1})^{\xi(\lambda_1)}\sigma_1\,(ac^{-1})^{\xi(\lambda_1\mu_1)}\ldots\rho_k\,(ca^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_k)}\sigma_k.$$

Express τ as $\tau^{(0)}\tau^{(1)}\tau^{(2)}$ where

$$\begin{aligned}\tau^{(1)} \;&=\; \rho_i\,(ca^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_i)}\sigma_i\,(ac^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_i\mu_i)}\\ &\qquad \rho_{i+1}\,(ca^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_{i+1})}\,\sigma_{i+1}\,(ac^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_{i+1}\mu_{i+1})}.\end{aligned}$$

Let

$$\bar{\tau}^{(1)} \;=\; \bar{\rho}\,(ca^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_{i-1}\mu_{i-1}\bar{\lambda})}\bar{\sigma}\,(ac^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_{i-1}\mu_{i-1}\bar{\lambda}\bar{\mu})}$$

where $\bar{\lambda}$ (resp. $\bar{\mu}$) is $\bar{v}$ with all $c^{\pm 1}$ and $d^{\pm 1}$ (resp. $a^{\pm 1}$ and $b^{\pm 1}$) removed. (So $\bar{\lambda} = \lambda_i\lambda_{i+1}$ and $\bar{\mu} = \mu_i\mu_{i+1}$.) It follows from Lemma 3.7 that $\bar{\tau} = \tau^{(0)}\bar{\tau}^{(1)}\tau^{(2)}$. So $\tau^{(1)} = \bar{\tau}^{(1)}$ in $F(a,b) \times F(c,d)$.

Obtain $\bar{\tau}$ from τ by changing the subword $\tau^{(1)}$ to $\bar{\tau}^{(1)}$ using Algorithm 1.

Lemma 4.2 *The cost of Algorithm* 2 *is at most* $136\left(\ell(v_iv_{i+1}) + |\xi(v_1\ldots v_{i-1})|\right)^2$.

Proof: The length of $\tau^{(1)}$ is at most

$$\begin{aligned}\ell(\rho_i\sigma_i\rho_{i+1}\sigma_{i+1}) + 2\Big(&|\xi(\lambda_1\mu_1\ldots\lambda_i)| \;+\; |\xi(\lambda_1\mu_1\ldots\lambda_i\mu_i)|\\ &+ |\xi(\lambda_1\mu_1\ldots\lambda_{i+1})| \;+\; |\xi(\lambda_1\mu_1\ldots\lambda_{i+1}\mu_{i+1})|\Big).\end{aligned}$$

As $\ell(\rho_i\sigma_i\rho_{i+1}\sigma_{i+1}) = 2\ell(\lambda_i\mu_i\lambda_{i+1}\mu_{i+1}) = 2\ell(v_iv_{i+1})$ and each of the four other terms differs from $|\xi(\lambda_1\mu_1\ldots\lambda_{i-1}\mu_{i-1})| = |\xi(v_1\ldots v_{i-1})|$ by at most $\ell(\lambda_i\mu_i\lambda_{i+1}\mu_{i+1})$,

$$\ell(\tau^{(1)}) \;\le\; 10\ell(v_iv_{i+1}) + 8|\xi(v_1\ldots v_{i-1})|.$$

Similarly,

$$\ell(\bar{\tau}^{(1)}) \;\le\; 6\ell(v_iv_{i+1}) + 4|\xi(v_1\ldots v_{i-1})|.$$

By Lemma 4.1 the cost of Algorithm 2 is at most $\ell(\tau^{(1)})^2 + \ell(\bar{\tau}^{(1)})^2$ — this then gives the (crude) estimate we claim. □

Our next subroutine merges two subwords in dyadic alternating form into one.

Algorithm 3 Merge two dyadic alternating subwords

Input: Two balanced words u and v, such that uv is a subword of $w = w(a, b, c, d, s)$, with τ_u and τ_v the dyadic alternating forms of u and v with respect to their positions in w.

Goal: Convert $\tau_u \tau_v$ to the dyadic alternating form τ of uv.

Method: Note that uv is balanced as it is the concatenation of two balanced words. Let $\hat{u}$, $\hat{v}$ and $\hat{w}$ be u, v and w respectively with all occurrences of the letters $s^{\pm 1}$ removed. Let $U, V \subset \mathbb{Z}$ be the sets of positions of the letters of $\hat{u}$ and $\hat{v}$ respectively in $\hat{w}$. Let $X_0, \ldots, X_n$ be a sequence of dyadic covers of $U \cup V$ as given by Corollary 3.3. Let σ_i be the partition of $\hat{u}\hat{v}$ corresponding to X_i and let τ_i be the partitioned alternating form of $\hat{u}\hat{v}$ with respect to σ_i. Then $\tau_0 = \tau_u \tau_v$ and $\tau_n = \tau$. For each i, convert τ_i to τ_{i+1} by calling Algorithm 2.

We postpone a full cost analysis of this algorithm to Section 6. In fact, there we will estimate the total cost of all the calls of Algorithm 2 throughout our main algorithm rather than their total cost within any single call on Algorithm 3. However, we will pause to give the following lemma which will be crucial to that analysis.

Lemma 4.3 *We continue with the notation of Algorithm 3. Fix i and say that the partition σ_{i+1} is formed from σ_i by combining the two subwords θ and ϕ. Then $\ell(\theta) = \ell(\phi)$ and either θ is a subword of $\hat{u}$ or ϕ is a subword of $\hat{v}$. Furthermore the cost of the call to Algorithm 2 converting τ_i to τ_{i+1} is at most $2176\,\ell(\theta)^2$.*

Proof: Say $X_i = \{S_1, \ldots, S_r\}$ and $X_{i+1} = \{S_1, \ldots, S_k \cup S_{k+1}, \ldots, S_r\}$, where the indexing on the S_i is ascending. Say that the partition σ_i of $\hat{u}\hat{v}$ is $\pi_1 \ldots \pi_r$. Then $\theta = \pi_k$ and $\phi = \pi_{k+1}$. By Corollary 3.3, $\ell(\pi_k) = \ell(\pi_{k+1})$ and $\pi_k \pi_{k+1}$ is not a subword of either $\hat{u}$ or $\hat{v}$.

By Lemma 4.2, the cost of the call to Algorithm 2 in question is at most $136(\ell(\pi_k\pi_{k+1}) + |\xi(\pi_1 \ldots \pi_{k-1})|)^2$. Note that $\pi_1 \ldots \pi_{k-1}$ is a subword of $\hat{u}$, which in turn is a subword of $\pi_1 \ldots \pi_{k+1}$, and thus

$$|\xi(\pi_1 \ldots \pi_{k-1}) - \xi(\hat{u})| \;\leq\; \ell(\pi_k \pi_{k+1}) \;=\; 2\ell(\pi_k).$$

Furthermore, since $\hat{u}$ is balanced, $\xi(\hat{u}) = 0$. Thus $|\xi(\pi_1 \ldots \pi_{k-1})| \leq 2\ell(\pi_k)$ and

$$136(\ell(\pi_k\pi_{k+1}) + |\xi(\pi_1 \ldots \pi_{k-1})|)^2 \;\leq\; 136(2\ell(\pi_k) + 2\ell(\pi_k))^2 \;\leq\; 2176\ell(\pi_k)^2.$$

□

Algorithm 4 Shuffling and canceling a $c^{\pm 1}$

Input: A word τ of the form

$$\rho_1\,(ca^{-1})^{\alpha_1}\sigma_1\,(ac^{-1})^{\beta_1}\ \ldots\ \rho_k\,(ca^{-1})^{\alpha_k}\sigma_k(ac^{-1})^{\beta_k}$$

where $\alpha_i, \beta_i \in \mathbb{Z}$, $\rho_i = \rho_i(a, b, c)$ and $\sigma_i = \sigma_i(a, c, d)$. Let $\epsilon \in \{\pm 1\}$ and define

$$\bar{\tau}\ =\ \rho_1\,(ca^{-1})^{\alpha_1+\epsilon}\sigma_1\,(ac^{-1})^{\beta_1+\epsilon}\ \ldots\ \rho_k\,(ca^{-1})^{\alpha_k+\epsilon}\sigma_k(ac^{-1})^{\beta_k+\epsilon}.$$

Goal: Convert $c^{\epsilon}\tau c^{-\epsilon}$ to $\bar{\tau}$.

Method: Working from left to right, shuffle c^{ϵ} through ρ_1, replace c^{ϵ} by $(ca^{-1})^{\epsilon}a^{\epsilon}$, shuffle a^{ϵ} through $(ca^{-1})^{\alpha_1}\sigma_1$, replace a^{ϵ} by $(ac^{-1})^{\epsilon}c^{\epsilon}$, and continue similarly. When c^{ϵ} emerges after $(ac^{-1})^{\beta_k}$ cancel it with the $c^{-\epsilon}$.

Lemma 4.4 *The cost of Algorithm 4 is at most* $2k + \ell(\tau)$.

Proof: Replacing c^{ϵ} by $(ca^{-1})^{\epsilon}a^{\epsilon}$ costs 1 when $\epsilon = -1$ and costs 0 otherwise. The same is true of replacing a^{ϵ} by $(ac^{-1})^{\epsilon}c^{\epsilon}$. The other contributions to cost stem from carrying a^{ϵ} or c^{ϵ} past letters of τ. $\square$

Our next subroutine expands a dyadic form subword by assimilating a letter on each side to produce a word in partitioned (but not necessarily dyadic) alternating form.

Algorithm 5 Assimilate x, y into a dyadic alternating word: I

Input: A subword of $w = w(a, b, c, d, s)$ of the form xvy where $x, y \in \{a, b, c, d\}^{\pm 1}$ have opposite exponents and $v = v(a, b, c, d, s)$ is balanced with dyadic alternating form τ. Write $\hat{v}$ and $\hat{w}$ for the words v and w respectively with all occurrences of the letter $s^{\pm 1}$ removed. Say $\hat{v}$ has dyadic partition $v_1 \ldots v_k$ in $\hat{w}$.

Goal: Convert $x\tau y$ into partitioned alternating form $\bar{\tau}$ with respect to the partition $xv_1 \ldots v_k y$.

Method: Using the notation of Definition 3.5,

$$\tau\ =\ \rho_1\,(ca^{-1})^{\xi(\lambda_1)}\sigma_1\,(ac^{-1})^{\xi(\lambda_1\mu_1)}\ldots\rho_k\,(ca^{-1})^{\xi(\lambda_1\mu_1\ldots\lambda_k)}\sigma_k.$$

Write x and y as $\lambda_0\mu_0$ and $\lambda_{k+1}\mu_{k+1}$, respectively, where λ_0 and λ_{k+1} are each in $\{a, a^{-1}, b, b^{-1}, \varepsilon\}$ and μ_0 and μ_{k+1} are each in $\{c, c^{-1}, d, d^{-1}, \varepsilon\}$.

Thus

$$\begin{aligned}\bar{\tau} &= \rho_0\,(ca^{-1})^{\xi(\lambda_0)}\sigma_0\,(ac^{-1})^{\xi(\lambda_0\mu_0)}\ldots\rho_{k+1}\,(ca^{-1})^{\xi(\lambda_0\mu_0\ldots\lambda_{k+1})}\sigma_{k+1}\\ &= \rho_0\,(ca^{-1})^{\xi(\lambda_0)}\sigma_0\,(ac^{-1})^{\xi(x)}\rho_1\,(ca^{-1})^{\xi(x)+\xi(\lambda_1)}\sigma_1\,(ac^{-1})^{\xi(x)+\xi(\lambda_1\mu_1)}\\ &\qquad\ldots\rho_{k+1}\,(ca^{-1})^{\xi(x)+\xi(\lambda_1\mu_1\ldots\lambda_{k+1})}\sigma_{k+1}.\end{aligned}$$

Transform $x\tau y$ to $\bar{\tau}$ by applying Algorithm 1 to convert

$$\begin{aligned}x &\mapsto \rho_0(ca^{-1})^{\xi(\lambda_0)}\sigma_0(ac^{-1})^{\xi(\lambda_0\mu_0)}c^{\xi(x)} \quad \text{and}\\ y &\mapsto c^{\xi(y)}\rho_{k+1}(ac^{-1})^{\xi(\mu_{k+1})}\sigma_{k+1},\\ &= c^{-\xi(x)}\rho_{k+1}(ca^{-1})^{\xi(x)+\xi(\lambda_1\mu_1\ldots\lambda_{k+1})}\sigma_{k+1},\end{aligned}$$

and then applying Algorithm 4 to the subword $c^{\xi(x)}\tau c^{-\xi(x)}$.

Lemma 4.5 *In $F(a,b)\times F(c,d)$,*

$$\begin{aligned}x &= \rho_0(ca^{-1})^{\xi(\lambda_0)}\sigma_0(ac^{-1})^{\xi(\lambda_0\mu_0)}c^{\xi(x)} \quad \text{and}\\ y &= c^{\xi(y)}\rho_{k+1}(ac^{-1})^{\xi(\mu_{k+1})}\sigma_{k+1},\end{aligned}$$

in the notation of Algorithm 5. Moreover, the cost of transforming x or y in this way is at most 2.

Proof: This is easily checked case-by-case. The reason the cost is so low is that most of the moves involved are free-expansions rather than applications of commutator relations. □

Lemma 4.6 *We continue with the notation of Algorithm 5. The total cost of calling this algorithm is at most $3\ell(\tau)+4$.*

Proof: By Lemmas 4.4 and 4.5 the cost is at most $2k+\ell(\tau)+4$. But $k\le\ell(\tau)$. □

The following routine builds on Algorithm 5. It assimilates a letter on either side of a dyadic subword to produce the dyadic form of the concatenated subword.

Algorithm 6 Assimilate x, y into a dyadic alternating word: II

Input: The partitioned alternating form $\bar{\tau}$ of xvy from Algorithm 5.

Goal: Convert the partitioned alternating form $\bar{\tau}$ of xvy from the previous algorithm to dyadic alternating form τ'.

Method: First apply Algorithm 5 to τ. Let $\bar{\tau}$ be the output.

Let $\hat{v}$ and $\hat{w}$ be the words v and w respectively with all occurrences of the letters $s^{\pm 1}$ deleted and let V be the set of positions of the letters of $\hat{v}$ in $\hat{w}$. Let $s = \min(V) - 1$ and $t = \max(V) + 1$. Thus s and t are the positions of the letters x and y respectively in $\hat{w}$.

Let $X_0, \ldots, X_n$ be a sequence of dyadic covers of $\{s\} \cup V$, as given by part 1 of Corollary 3.4, converting $X_0 = \{\{s\}\} \cup \mathrm{MDC}(V)$ to $X_n = \mathrm{MDC}(\{s\} \cup V)$. For each i, define Y_i to be $X_i \cup \{\{t\}\}$. Thus $Y_0, \ldots, Y_n$ is a sequence of dyadic covers of $\{s\} \cup V \cup \{t\}$ converting $Y_0 = \{\{s\}\} \cup \mathrm{MDC}(V) \cup \{\{t\}\}$ to $Y_n = \mathrm{MDC}(\{s\} \cup V) \cup \{\{t\}\}$.

Let $Y_n, \ldots, Y_m$ be a sequence of dyadic covers of $\{s\} \cup V \cup \{t\}$, as given by part 2 of Corollary 3.4, converting, converting $Y_n = \mathrm{MDC}(\{s\} \cup V) \cup \{\{t\}\}$ to $Y_m = \mathrm{MDC}(\{s\} \cup V \cup \{t\})$.

For each $i = 0, \ldots, m$, let σ_i be the partition of $x\hat{v}y$ corresponding to Y_i, and let τ_i be the partitioned alternating form of $x\hat{v}y$ with respect to σ_i. Then $\tau_0 = \bar{\tau}$ and $\tau_m = \tau'$. For each i, call Algorithm 2 to convert τ_i to τ_{i+1}.

A lemma analogous to Lemma 4.3 will be important when we come to analyse cost.

Lemma 4.7 *We continue with the notation of Algorithm* 6*. Fix i and say the partition σ_{i+1} of $x\hat{v}y$ is formed from σ_i by combining the subwords θ and ϕ. Then $\ell(\theta) = \ell(\phi)$ and the cost of the call to Algorithm* 2 *which converts τ_i to τ_{i+1} is at most* $2176\ell(\theta)^2$.

Proof: First consider the case that $0 \leq i \leq n-1$. Say that the partition σ_i of $x\hat{v}y$ is $\pi_1 \ldots \pi_k$. Then $\theta = \pi_1$ and $\phi = \pi_2$ and the partition σ_{i+1} is $\bar{\pi}\pi_3 \ldots \pi_k$, where $\bar{\pi} = \theta\phi$. By part 1 of Corollary 3.4, $\ell(\theta) = \ell(\phi)$ and by Lemma 4.2 the cost of the call to Algorithm 2 is at most

$$136\ell(\theta\phi)^2 \;=\; 136(2\ell(\theta))^2 \;=\; 544\ell(\theta)^2.$$

Now consider the case that $n \leq i \leq m-1$. Say that the partition σ_i of $x\hat{v}y$ is $\pi_1 \ldots \pi_k$. Then $\theta = \pi_{k-1}$ and $\phi = \pi_k$ and the partition σ_{i+1} is $\pi_1 \ldots \pi_{k-1}\bar{\pi}$ where $\bar{\pi} = \pi_{k-1}\pi_k$. By part 2 of Corollary 3.4, $\ell(\theta) = \ell(\phi)$ and by Lemma 4.2 the cost of the call to Algorithm 2 is at most

$$\begin{aligned}
136(\ell(\theta\phi) + |\xi(\pi_1 \ldots \pi_{k-1})|)^2 &= 136(2\ell(\theta) + |\xi(\theta\phi)|)^2 \\
&\leq 136(2\ell(\theta) + \ell(\theta\phi))^2 \\
&= 136(2\ell(\theta) + 2\ell(\theta))^2 \\
&= 2176\ell(\theta)^2.
\end{aligned}$$

□

5 Our main algorithm

We are now ready to give our main algorithm, which converts a null-homotopic word of length n in S to ε via $n/2$ intermediate words, each of length at most $10n$, by applying relations from the presentation (2).

Algorithm 7 Our main algorithm

Input: A word $w = w(a, b, c, d, s)$ representing 1 in S

Goal: Reduce w to ε by applying defining relators of S.

Method: Define $n := \ell(w)$. As all the defining relators in the presentation (2) of S are of even length, n is even. We will obtain a sequence of words w_i and a sequence T_i of subsets of $\{1, \ldots, n\}$ beginning with $w_0 := w$ and $T_0 := \emptyset$.

In fact, w and T_i will define w_i as follows. Express w as

$$w \quad = \quad u_1 v_1 u_2 \ldots v_{k-1} u_k, \tag{3}$$

where T_i is the set of positions of the letters of the v_j in w and $v_1, u_2, \ldots, u_{k-1}, v_{k-1} \neq \varepsilon$. One sees inductively from the construction of successive T_i below that the v_j are balanced. Let τ_j be the dyadic alternating form of v_j in w. Then

$$w_i \quad = \quad u_1 \tau_1 u_2 \ldots \tau_{k-1} u_k. \tag{4}$$

We will now explain how to obtain T_{i+1} from T_i and then how to use relations in S to transform w_i, expressed as (4), to w_{i+1}.

As w and the v_j are all balanced, $u_1 u_2 \ldots u_k$ is also balanced by Lemma 2.5. So Lemma 2.6 applies and tells us there is a subword xy in $u_1 \ldots u_n$ that either equals $s^{\pm 1} s^{\mp 1}$ or is such that x and y are in $\{a, b, c, d\}^{\pm 1}$ and have opposite exponents. Add the positions of x and y in w to T_i to obtain T_{i+1}.

In w_i these x and y are either adjacent or separated by some τ_j.

If $xy = s^{\pm 1} s^{\mp 1}$, then remove x and y by shuffling x through τ_j (if present) and cancelling it with y.

If $x, y \in \{a, b, c, d\}^{\pm 1}$, then apply Algorithm 6 to replace $x \tau_j y$ with the dyadic alternating form τ' of the subword $x v_j y$ of w.

Next, if (in either case) we have brought two or three dyadic alternating form subwords together (that is, either $u_j = x$ and $j \neq 1$, or $u_{j+1} = y$ and $j + 1 \neq k$, or both) then merge them using Algorithm 3 (once or twice, as necessary). The result is the word w_{i+1}.

After $n/2$ iterations, $T_{n/2} = \{1, \ldots, n\}$ and $w_{n/2}$ contains no $s^{\pm 1}$ and represents 1 in $F(a,b) \times F(c,d)$. Reduce $w_{n/2}$ to ε using Algorithm 1.

---

6 Cost analysis

We will estimate the cost of our main algorithm in terms of $n = \ell(w)$. Cost is incurred in four ways: shuffling an $s^{\pm 1}$ to be cancelled with an $s^{\mp 1}$, calls of Algorithm 3, calls of Algorithm 6, and converting $w_{n/2}$ to ε at the end.

Shuffling an $s^{\pm 1}$ to be cancelled with an $s^{\mp 1}$ costs at most $5n$ because the τ_j between them (if present) is alternating and so $\ell(\tau_j)/2$ relations are required, and $\ell(\tau_j)/2 \leq 5\ell(v_j) \leq 5n$ by Lemma 3.11. Since we do such shuffling at most $n/2$ times, this contributes no more than $5n^2/2$ to the total cost.

The final step, converting $w_{n/2}$ to ε, costs at most $\ell(w_{n/2})^2 \leq 100n^2$ by Lemmas 3.11 and 4.1.

Consider the call to Algorithm 6 converting $x\tau_j y$ to the dyadic alternating form of xv_jy. The cost of this call can be divided into the cost of calls to Algorithm 5 and the cost of calls to Algorithm 2. By Lemma 4.6 each call to Algorithm 5 costs at most $3\ell(\tau_j) + 4$ and by Lemma 3.11 we have that $\ell(\tau_j) \leq 10n$. Thus the cost of each call to Algorithm 5 is at most $30n + 4$. In total Algorithm 6, and hence Algorithm 5, is called at most $n/2$ times and so the total cost of all calls to Algorithm 5 is at most $15n^2 + 2n$.

This leaves just the calls to Algorithm 2 to consider. We will bound the total cost of all calls to this algorithm arising from either Algorithm 3 or Algorithm 6.

Let $\hat{w}$ be w with all $s^{\pm 1}$ deleted. Recall that $D_{r,j} = \mathbb{Z} \cap [j2^r, (j+1)2^r)$. For integers $r \geq 0$ and $j \geq 1$, define $v_{r,j}$ to be the subword of $\hat{w}$ whose letters are in positions $D_{r,j}$ in $\hat{w}$.

By Lemmas 4.3 and 4.7, each call on Algorithm 2 costs at most $2176\,\ell(v_{r,j})^2 = 2176\,.\,(2^r)^2$ for some r and j, where $v_{r,j}$ is the θ of those lemmas. The key point is that this pair (r,j) never occurs in this way in any other call on Algorithm 2 — once that merge has been made it is never repeated. So the total cost of all applications of Algorithm 2 is at most $2176 \sum_{\mathcal{D}} 2^{2r}$, where $\mathcal{D}$ is the set of all pairs (r,j) such that $D_{r,j} \subseteq \{1, \ldots, \ell(\hat{w})\}$, and as $\ell(\hat{w}) \leq n$,

$$\sum_{\mathcal{D}} 2^{2r} \;\leq\; \sum_{r=0}^{\lceil \log_2 n \rceil} \frac{n}{2^r} 2^{2r} \;\leq\; n \sum_{r=0}^{\lceil \log_2 n \rceil} 2^r \;\leq\; n\left(2^{1+\lceil \log_2 n \rceil} - 1\right) \;\leq\; n(4n-1).$$

Summing our estimates $5n^2/2$, $100n^2$, $15n^2 + 2n$ and $2176n(4n-1)$, we get the upper bound on cost of $17643n^2/2 - 2174n$, which establishes our theorem.

References

[1] G. Baumslag, M. R. Bridson, C. F. Miller, and H. Short. Finitely presented subgroups of automatic groups and their isoperimetric functions. *J. London Math. Soc. (2)*, 56(2):292–304, 1997.

[2] R. Bieri. Homological dimension of discrete groups. Queen Mary Lecture Notes, 1976.

[3] M. R. Bridson. personal communication.

[4] M. R. Bridson. Doubles, finiteness properties of groups, and quadratic isoperimetric inequalities. *Journal of Algebra*, 214:652–667, 1999.

[5] S. M. Gersten. Finiteness properties of asynchronously automatic groups. In R. Charney, M. Davis, and M. Shapiro, editors, *Geometric Group Theory*, volume 3 of *Ohio State University, Mathematical Research Institute Publications*, pages 121–133. de Gruyter, 1995.

[6] D. Groves. personal communication.

[7] P. Papasoglu. On the asymptotic invariants of groups satisfying a quadratic isoperimetric inequality. *J. Differential Geom.*, 44:789–806, 1996.

[8] T. R. Riley. Higher connectedness of asymptotic cones. *Topology*, 42:1289–1352, 2003.

[9] J. Stallings. A finitely presented group whose 3-dimensional integral homology is not finitely generated. *Amer. J. Math.*, 85:541–543, 1963.