Ordering the braid groups

Roger Fenn, Michael T Greene, Dale Rolfsen
Colin Rourke, Bert Wiest

Email: R.A.Fenn@sussex.ac.uk Michael.Greene@uk.radan.com
rolfsen@math.ubc.ca cpr@maths.warwick.ac.uk
bertw@gyptis.univ-mrs.fr

Abstract

We give an explicit geometric argument that Artin's braid group B_n is right-orderable. The construction is elementary, natural, and leads to a new, effectively computable, canonical form for braids which we call *left-consistent canonical form*. The left-consistent form of a braid which is positive (respectively negative) in our order has consistently positive (respectively negative) exponent in the smallest braid generator which occurs. It follows that our ordering is identical to that of Dehornoy [6], constructed by very different means, and we recover Dehornoy's main theorem that any braid can be put into such a form using either positive or negative exponent in the smallest generator but not both.

Our definition of order is strongly connected with Mosher's normal form [13] and this leads to an algorithm to decide whether a given braid is positive, trivial, or negative which is quadratic in the length of the braid word.

AMS Classification numbers Primary: 20F60, 06F15, 20F36

Secondary: 57M07, 57M25

Keywords: Braid, right-invariant order, left-consistent canonical form, quadratic time algorithm, cutting sequence

0 Introduction

Dehornoy [5, 6, 7] has proved that the braid group is right-orderable. More precisely, there is a total order on the elements of the braid group B_n which is right invariant in the following sense. Suppose that $\alpha,\ \beta,\ \gamma \in B_n$ and $\alpha < \beta$, then $\alpha\gamma < \beta\gamma$. This ordering is uniquely defined by the condition that a braid $\beta_0 \sigma_i \beta_1$ is positive (ie greater than the identity braid), where β_0, β_1 are words in $\sigma_{i+1}^{\pm 1}, \ldots, \sigma_{n-1}^{\pm 1}$. Dehornoy's proof is based on some highly complicated algebra connected with left-distributive systems. In this paper we construct this order geometrically using elementary arguments.

Our construction leads to a new, effectively computable, canonical form for braids which we call *left-consistent canonical form*. The left-consistent form of a positive braid has the general shape

$$\beta_0 \sigma_i^e \beta_1 \ldots \beta_{l-1} \sigma_i^e \beta_l$$

where the β_i are words in $\sigma_{i+1}, \ldots, \sigma_{n-1}$ and their inverses, and $e = +1$. For a negative braid the form is similar but with $e = -1$. It follows at once that our ordering is identical to Dehornoy's and we recover Dehornoy's main theorem that any braid can be put into such a shape for $e = 1$ or $e = -1$ but not both.

Our definition of order is strongly connected with Mosher's automatic structure [13] and this implies that the braid group is *order automatic*, ie the order can be detected from the automatic normal form by a finite state automaton. Furthermore the resulting algorithm to decide whether a given braid is positive, trivial, or negative is linear in the length of the Mosher normal form and hence quadratic in the length of the braid word (in contrast, Dehornoy's algorithm [7], although apparently fast in practice is only known to be exponential).

The paper is organised as follows. Section 1 contains basic definitions and introduces the curve diagram associated to a braid. In section 2 we prove that a curve diagram can be placed in a unique reduced form with respect to another and in section 3 we define the order by comparing the two curve diagrams in reduced form, and prove that it is right-invariant. In section 4 we construct the left-consistent canonical form of a braid, deduce that our order coincides with Dehornoy's and recover Dehornoy's results. In section 5 we give some counterexamples connected with the order, and in section 6 we make the connection with Mosher's normal form and deduce the existence of the quadratic time algorithm to detect order. Finally, in an appendix, we use *cutting sequences* to give a formal algorithm to turn a braid into the new left-consistent canonical form; note that this algorithm is not quadratic time.

Acknowledgements We are grateful to the organisers of the low-dimensional topology conference held at the Isle of Thorns in Spring 1997, which was supported by the LMS, for providing a congenial atmosphere for the initial work on this paper. We are also grateful to Caroline Series for suggesting that our curve diagrams might be related to Mosher's normal form for mapping class groups. We would also like to thank the referee for helpful comments and a speedy report. Bert Wiest is supported by a TMR (Marie Curie) research training grant.

1 Braids and curve diagrams

Let D^2 be the closed unit disk in $\mathbb{C}$, and let D_n be the disk D^2 with n distinct points in the real interval $(-1,1)$ removed. We consider the group B_n of self-homeomorphisms $\gamma\colon D_n \to D_n$ with $\gamma|_{\partial D_n} = id$, up to isotopy of D_n fixed on ∂D_n. Multiplication in B_n is defined by composition. The group B_n is well-defined independently of the points removed; indeed if D'_n is a disk with any n–tuple of points removed and B'_n the corresponding group then there is an isomorphism $B'_n \cong B_n$; if these points also lie on $(-1,1)$ then this isomorphism is natural.

The group B_n is isomorphic to the group $\widetilde{B}_n$ of braids on n strings, with multiplication given by concatenation: if α, β are braids (pictured vertically) then $\alpha \cdot \beta$ is α above β. It is well known that this group has presentation

$$\widetilde{B}_n \cong \langle \sigma_1, \ldots, \sigma_{n-1} \mid \sigma_i\sigma_j = \sigma_j\sigma_i \text{ if } |i-j| \geqslant 2,\ \sigma_{i+1}\sigma_i\sigma_{i+1} = \sigma_i\sigma_{i+1}\sigma_i \rangle,$$

where the generator σ_i $(i \in \{1, \ldots n-1\})$ is indicated in figure 1.

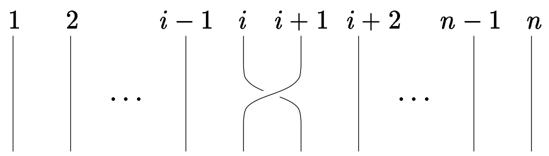

Figure 1: The standard generator σ_i of the braid group on n strings

The isomorphism $\widetilde{B}_n \to B_n$ is given by 'putting the braid in a solid cylinder and sliding D_n once along it'. The inverse map is defined as follows: extend a given homeomorphism $\gamma\colon D_n \to D_n$ to a homeomorphism $\gamma'\colon D^2 \to D^2$, then find a boundary-fixing isotopy $\gamma_t\colon D^2 \to D^2$ with $\gamma_0 = id$ and $\gamma_1 = \gamma$. Then the flow of the n holes of D_n under γ_t defines a braid on n strings. For details see [2].

On D_n we draw $n+1$ line segments as in figure 2(a). If γ is a homeomorphism of D_n representing an element $[\gamma]$ of B_n, then γ sends these line segments to $n+1$ disjoint embedded curves, and if $[\gamma_1] = [\gamma_2] \in B_n$ then γ_1 and γ_2 give rise to isotopic collections of curves. For instance, figure 2 shows the effect of the braid $\sigma_1\sigma_2^{-1} \in B_3$. Here the holes of D_n, as well as ± 1 are indicated by black dots. We call such a diagram of $n+1$ disjoint simple curves in an n–punctured disk a *curve diagram*, and we number the curves in the diagram 1 to $n+1$, as in figure 2.

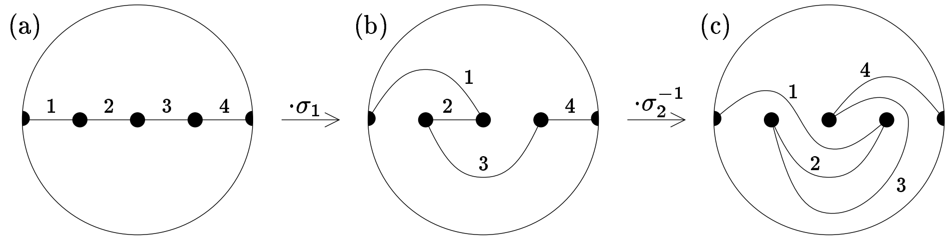

Figure 2: Curve diagrams of the braids 1, σ_1, and $\sigma_1\sigma_2^{-1}$

Conversely, from the curve diagram we can reconstruct the homeomorphism γ up to boundary-fixing isotopy; that is, we can reconstruct the element of the braid group.

2 Reduced form

Let Γ and Δ be curve diagrams of two braids γ and δ, say. In order to compare Γ and Δ, we superimpose the two diagrams and *reduce* the situation by removing unnecessary intersections. This process is well known and often called "pulling tight" (see eg [13]).

We will denote the ith curve of a curve diagram such as Γ by Γ_i. The curves Γ_i and the Δ_j are called *parallel* if they connect the same pairs of points and are isotopic in D_n. For instance, curve 3 of figure 2(b) and curve 2 of figure 2(c) are parallel. We define Δ to be *transverse* to Γ if every curve of Δ either coincides precisely with some (parallel) curve of Γ, or intersects the curves of Γ transversely.

We define the *intersection index* of two transverse curve diagrams to be

$$n + 1 + \#(\text{transverse intersections}) - \#(\text{coincident curves}).$$

(The geometric significance is that D_n cut along Γ has two components, and cutting *in addition* along Δ increases the number of components by the intersection index.) For example, the diagrams in figure 2(a) and 2(c) have intersection index 6, the diagrams in figure 2(a) and 2(b) have intersection index 2 and the diagrams in figure 2(b) and 2(c) have intersection index 5. The intersection index of two curve diagrams is 0 if and only if the diagrams are identical.

We now fix a curve-diagram Γ for γ, and look at all possible curve-diagrams for δ. They are all isotopic in D_n, but they may have very different intersection-indices with Γ. We say Δ and Δ' are *equivalent* (with respect to Γ) if they are related by an isotopy of D_n, which is fixed on curves of Δ which coincide with curves of Γ, and which leaves the diagrams transverse all the time. So coincident curves remain coincident, and the intersection index remains unchanged.

We define a *D–disk*[1] *between Δ and Γ* to be a subset of D_n homeomorphic to an open disk, which is bounded by one open segment of some curve of Δ, one open segment of some curve of Γ, and two points, each of which may be an intersection-point of the two curves or one of the 'holes' of D_n, or $\pm 1 \in D_n$. three types of D–disks (types (a), (b), and (c)), indicated in figure 3, where the curve-diagram Γ is drawn with dashed, and Δ with solid lines, and the dots denote holes or ± 1.

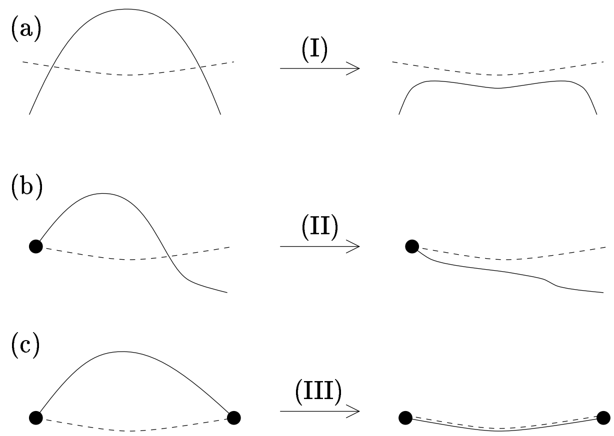

Figure 3: Three types of D–disks, and how to use them to reduce intersection indices

If there are no D–disks between Δ and Γ then we say Δ and Γ are *reduced*. If the curve-diagrams Δ and Γ are not reduced, ie if they have a D–disk, then we can isotope Δ so as to reduce the intersection index with Γ (figure 3).

[1] D–disks are often called "bigons" in the literature.

This isotopy consists of 'sliding a segment of a curve of Δ across the D–disk' (for reduction moves (I) and (II)), and of 'squashing a D–disk to a line' (for reduction move (III)). The three moves reduce the intersection index by 2, 1, 1, respectively. We observe that any curve-diagram Δ with intersection index 0 with Γ is reduced. Thus we can reduce curve diagrams by a finite sequence of 'isotopies across D–disks' as in figure 3.

Lemma 2.1 (Triple reduction lemma) *Suppose Σ, Γ and Δ are three curve diagrams such that Γ and Δ are both reduced with respect to Σ. Then there exists an isotopy between Δ and a curve diagram Δ', which is an equivalence with respect to Σ, such that Σ, Γ and Δ' are pairwise reduced.*

Proof We consider a D–disk bounded by one segment of curve of Γ and one of Δ. This D–disk may have several intersections with Σ. There are, a priori, three possibilities for the type of such an intersection — they are indicated in figure 4, labelled (1), (2), and (3). (In this figure, the D–disk is of type (b), the cases of types (a) and (c) are similar.)

However, (1) and (2) are impossible, because Γ and Δ are reduced with respect to Σ. So all intersections are of type (3), and the D–disk can be removed without disturbing the reduction of Σ with respect to Γ or Δ, as indicated in figure 4. The statement follows inductively. $\square$

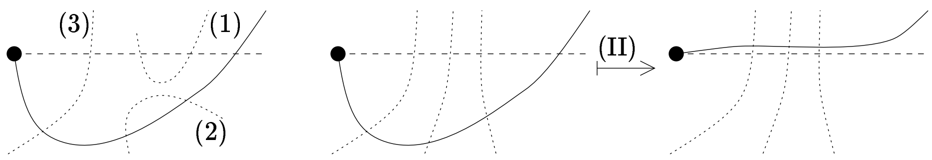

Figure 4: The solid line is Δ, the dashed Γ, and the dotted Σ

Lemma 2.2 *If two isotopic curve diagrams Γ and Δ are reduced with respect to each other, then they coincide.*

Proof Suppose that the first curve Γ_1 of Γ does not coincide with the first curve Δ_1 of Δ. Consider the word obtained by reading the intersections of Γ_1 with the curves of Δ in order. Since Γ_1 is isotopic to Δ_1, this word must cancel to the trivial word. It follows by a simple innermost disk argument that there must be a D–disk. Hence Γ_1 must coincide with Δ_1. Similarly all curves of Γ and Δ must coincide. $\square$

Proposition 2.3 *If two curve diagrams Δ and Δ' of a braid δ are reduced with respect to Γ then they are equivalent with respect to Γ.*

Proof By the triple reduction lemma we may reduce Δ with respect to Δ' by an isotopy of Δ which is an equivalence with respect to Γ. After this reduction Δ and Δ' coincide by lemma 2.2. $\square$

We have proved that by reducing a curve diagram Δ with respect to a curve diagram Γ we can bring Δ into a uniquely defined standard form with respect to Γ. In particular reduction of Δ with respect to the trivial curve diagram representing $1 \in B_n$ (figure 2(a)) leads to a canonical representation of braids in terms of *cutting sequences*, which will be discussed in detail in the appendix.

Remark 2.4 The following observation will be crucial at a later point. Let Γ and Δ be transverse curve diagrams. Suppose the curve Δ_i on its own is reduced with respect to Γ. Then we can reduce Δ with respect to Γ by an isotopy of Δ which is fixed on Δ_i.

3 The right-invariant order on B_n

We define a total ordering on the braid group B_n as follows. Suppose γ and δ are two braids on n strings. We let Γ be a curve diagram for γ, and Δ be a curve diagram for δ which is reduced with respect to Γ. The collection of curves of Γ cuts D_n into two components, which we call the *upper* and the *lower* component, containing the points $\sqrt{-1}$ respectively $-\sqrt{-1}$ in $D_n \subseteq \mathbb{C}$. We orient the curves of Δ coherently such that we obtain a path starting at $-1 \in D_n$ and ending at $1 \in D_n$.

If all curves of Δ coincide with the corresponding curves of Γ then the braids γ and δ are equal. Suppose that curves $1, 2, \ldots, i-1$ of Δ agree with the corresponding curves of Γ, and the ith is the first transverse one, $1 \leq i \leq n+1$. This oriented curve has the same startpoint as the ith curve of Γ, and first branches off Γ either into the upper or the lower component. In the first case we define $\delta > \gamma$, in the second $\delta < \gamma$. This is well-defined, by proposition 2.3.

Example All the diagrams in figure 2 are reduced with respect to each other, and we observe that $1 < \sigma_1 \cdot \sigma_2^{-1} < \sigma_1$.

Proposition 3.1 *The relation '$<$' is an ordering, ie if σ, γ, δ are braids with $\sigma < \gamma < \delta$ then $\sigma < \delta$.*

Proof By the triple reduction lemma 2.1 we can find curve diagrams Σ, Γ, Δ of these braids which are all pairwise reduced. The statement of the proposition follows immediately: if Γ branches off Σ to the left and Δ branches off Γ to the left, then Δ branches off Σ to the left. □

Proposition 3.2 *The ordering '$<$' is right invariant.*

Proof Suppose we have two braids δ and γ with $\delta < \gamma$, and with reduced curve diagrams Δ and Γ. Let σ be a further braid, ie a homeomorphism of D_n fixing ∂D_n. We obtain the curve diagrams for $\delta \cdot \sigma$ and $\gamma \cdot \sigma$ by applying σ to Δ and Γ. The resulting curve diagrams $\sigma(\Delta)$ and $\sigma(\Gamma)$ are still reduced, and $\sigma(\Delta)$ still branches off $\sigma(\Gamma)$ into the lower component of $D_n \setminus \sigma(\Gamma)$, so $\delta \cdot \sigma < \gamma \cdot \sigma$. □

Let ϵ be the trivial braid, with standard curve diagram E (see figure 2(a)). We call a braid γ *positive* if $\gamma > \epsilon$, and *negative* if $\gamma < \epsilon$. If we want to stress that the first $i-1$ curves of Γ are parallel to the corresponding curves of E, and the ith is the first non-parallel one, then we say γ is *i–positive* respectively *i–negative*. Since there is a very similar concept of σ_i–positive (see the next section) we shall often say *geometrically* i–positive or negative. Given two braids γ and δ such that $\gamma > \delta$ we say γ is (geometrically) *i–greater* than δ if the ith curves are the first non-parallel ones. Any curve diagram in which the first $i-1$ curves are parallel to the corresponding curves of E is called *$(i-1)$–neutral*.

We note some simple consequences of right invariance. We have $\gamma > \epsilon$ if and only if $\epsilon > \gamma^{-1}$, so the inverse of a positive braid is negative. If $\gamma > \epsilon$ and δ is any braid, then $\gamma\delta > \delta$. (*Warning:* it need not be true that $\delta\gamma > \delta$ — see the next section.) In particular, the product of positive braids is positive.

4 Left-consistent canonical form

In this section we connect our ordering with Dehornoy's [5]. The following definition is taken from [5]. A word of the form

$$\beta_0 \sigma_i \beta_1 \sigma_i \ldots \sigma_i \beta_k,$$

where $i \in \{1, \ldots, n-1\}$, and $\beta_0, \ldots \beta_k$ are words in the letters $\sigma_{i+1}^{\pm 1} \ldots \sigma_{n-1}^{\pm 1}$ is called a *σ_i–positive* word. A braid is *σ_i–positive* if it can be represented by a σ_i–positive word. A braid is called *σ_i–negative* if its inverse is σ_i–positive. We shall say that a braid is σ–positive or negative if it is σ_i–positive or negative for some i. The following is the main result from [5]:

Dehornoy's theorem *Every braid is precisely one of the following three: σ–positive, or σ–negative, or trivial.*

Dehornoy uses this theorem to define a right-invariant order by $\alpha > \beta \iff \alpha\beta^{-1}$ is σ–positive. We shall prove that this order coincides with the order we defined in the last section by showing that the concepts of geometrically i–positive and σ_i–positive coincide. One way is easy.

Proposition 4.1 *A braid which is σ_i–positive is geometrically i–positive.*

Proof A braid which can be represented by a word $\beta\sigma_i$, where β is a word in the letters $\sigma_{i+1}, \ldots, \sigma_{n-1}$, is geometrically i–positive. To see this think of the homeomorphism determined by the braid word as a sequence of twists of adjacent holes around each other: β leaves the first i curves untouched and then σ_i twists the ith hole around the $(i+1)$st producing a curve diagram in which the ith curve moves into the upper half of D_n. Now by definition, every Dehornoy positive braid is a product of such words. The proposition now follows from the fact that the product of two geometrically i–positive braids is again geometrically i–positive. □

The proposition immediately implies part of Dehornoy's theorem: every braid can take *at most* one of the three possible forms. To complete the proof that the concepts of geometrically i–positive and σ_i–positive coincide and to recover the remainder of Dehornoy's theorem we shall construct a canonical σ_i–positive form for a given geometrically i–positive braid. This is the *left-consistent canonical form* of the braid:

Theorem 4.2 (Left-consistent canonical form) *Let γ be a geometrically i–positive braid. Then there is a canonically defined σ_i–positive word which represents the same element of B_n.*

We define the *complexity* of a braid γ as follows. Take a curve diagram Γ for γ which is reduced with respect to the trivial curve diagram E. Suppose that the first $j-1$ curves of Γ coincide with the first $j-1$ curves of E and that the jth curve does not. Let $m \geq 0$ be the number of transverse intersections of Γ with jth curve of E. The *complexity of* γ is the pair (j, m). We order complexity lexicographically with j in reverse order. Thus $(1, m)$ is more complex than $(2, n)$ for any m, n whilst (j, m) is more complex than (j, n) if and only if $m > n$. The main step in the proof of theorem 4.2 is the following:

Proposition 4.3 *Suppose that γ is a geometrically i–positive braid. Then there is a word β in the braid generators $\sigma_i, \ldots, \sigma_{n-1}$ and their inverses such that*

(1) *β contains σ_i^{-1} exactly once*

(2) *β does not contain σ_i*

(3) *$\gamma\beta$ is either geometrically i–positive or geometrically i–neutral*

(4) *$\gamma\beta$ has smaller complexity than γ.*

Furthermore there is a canonical choice for β.

Theorem 4.2 follows from proposition 4.3 by induction on complexity because, by (4) and induction, $\gamma' := \gamma\beta$ has a canonical form which by (3) is either σ_i–positive or σ_j–positive or negative for $j > i$ and then $\gamma'\beta^{-1}$ is the canonical form for γ.

Proof of proposition 4.3 For definiteness we shall deal with the case $i = 1$ first. (We shall see that the general case is essentially the same as this case.) So let γ be geometrically 1–positive braid and Γ a curve diagram for γ which is reduced with respect to the trivial curve diagram. We shall define β geometrically by sliding one particular hole of D_n along a *useful arc.*

Let $E_1 \subseteq D_n$ be the 1st curve of E, ie a straight line from -1 to the leftmost hole of D_n, excluding this hole. We define a *useful arc* to be a segment b of some curve of Γ starting at some point of E_1 (possibly -1), and ending at some hole of D_n other than the leftmost one such that

- the interior of b does not intersect E_1,
- an initial segment of the arc b lies in the upper half of the disk, ie the intersection of a neighbourhood of E_1 with the interior of b consists of a line segment in the upper component of $D_n \setminus E$.

Suppose that Γ contains useful arcs. Then each of them has precisely one point of intersection with E_1 and we call the one whose intersection point is leftmost the *leftmost useful arc.* Let b be the leftmost useful arc. If b starts in the interior of E_1, then we can slide the hole of D_n at the endpoint of b along b and back into E_1. If b starts at -1, then we push a small initial segment of b into E_1, and then perform the slide of the hole of D_n (see figure 5 where b is dotted). In either case we obtain a curve diagram Γ' representing a braid γ'. Now Γ' need not be reduced with respect to E. But notice that γ' has lower

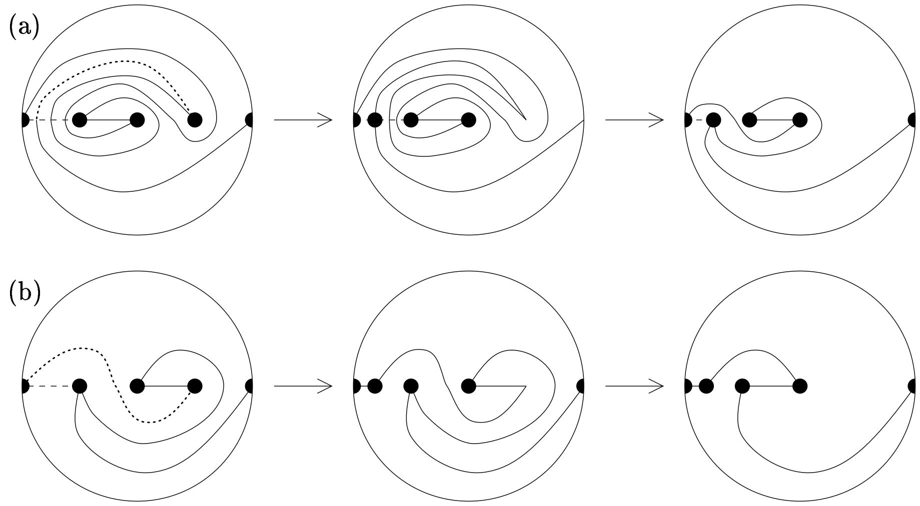

Figure 5: Slide of a hole along a useful arc, followed by a reduction

complexity than γ since the new E_1 now stops at the intersection of b with the old E_1 and hence there are fewer intersections with Γ' even before reduction.

The movement of the hole of D_n along b defines a braid β on n strings, with $\gamma' = \gamma\beta$. Furthermore we can decompose β as a canonical word in the generators σ_j by writing down the appropriate σ_j or σ_j^{-1} whenever the hole passes over or under another hole. But, by definition of useful arc, the hole only passes once over or under the leftmost hole and it passes over and to the left and hence the word that we read contains σ_1^{-1} only once and does not contain σ_1.

Therefore to prove case $i = 1$ of proposition 4.3 it remains to prove the following two claims:

Claim 1 The diagram contains a useful arc.

Claim 2 The diagram Γ' obtained by sliding a hole of D_n along the leftmost useful arc is either 1–positive or 1–neutral, but not 1–negative.

To prove claim 1, we consider the first curve of Γ starting at -1. If it ends in a hole other than the leftmost one and does not intersect E_1 then it is a useful arc (figure 6(a)). Otherwise we consider the closed curve in D^2 starting at -1, along the first curve of Γ, up to its first intersection with the closure of E_1 in D^2, and then back in a straight line to the point -1. This curve bounds a disk S in D^2, which may be of three different types: Γ hits E_1 either from above, or from below, or in the leftmost hole of D_n (see figure 6(b),(d),(c)). In cases

(b) and (c) we note that since Γ and E are reduced, at least one hole of D_n must lie in the interior of S. Moreover, all holes of D_n are connected by curves of Γ, so there exists a curve of Γ connecting one of the holes in S to one of the holes outside S or the point $1 \in D_n$. The first component of the intersection of this curve with S is a useful arc.

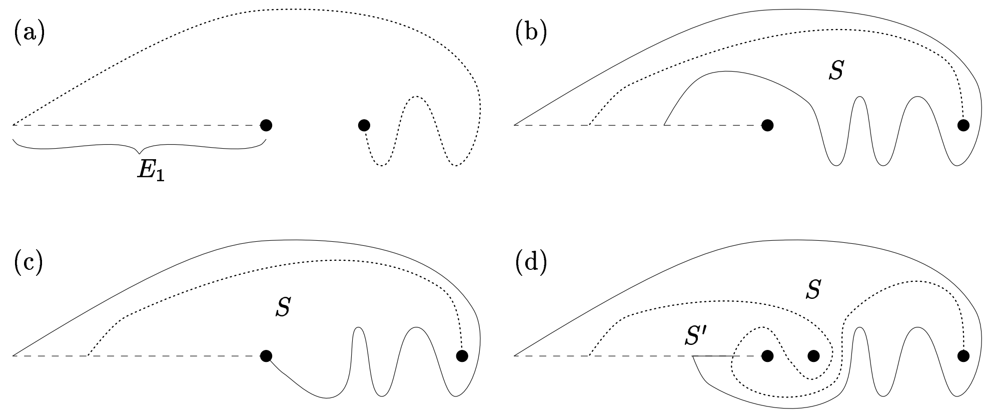

Figure 6: How to find a useful arc

In case (d) we walk along the oriented curve in D^2 starting at -1, along the curves of Γ. We write down the symbol $+$ whenever we hit E_1 from below (or at -1), and $-$ if we hit E_1 from above or in the leftmost hole of D_n. The sequence starts with a $+$, and since the curve has to leave the disk S it must contain a $-$. It follows that the string $+-$ must occur in the sequence; it represents an arc which, together with a segment of E_1, bounds a disk S' in D^2. See figure 6(d): S' is bounded by part of the dotted arc between two intersections with E_1 and part of E_1. Since Γ and E are reduced, S' contains a hole other than the leftmost one in its boundary or in its interior. In the first case, a segment of top (dotted) boundary of S' is a useful arc; in the second case the disk S' is of the type indicated in figure 6(b) or (c), so there is a useful arc inside S'. This finishes the proof of claim 1.

To prove claim 2, we distinguish two cases: either the leftmost useful arc b starts at the point -1, or it starts at some point in the interior of E_1. In the first case (eg figure 5(b)), the curve diagram Γ' obtained by sliding a hole along b to near -1 is 1–neutral.

In the second case (figure 5(a)) the curve diagram Γ' is 1–positive, as we now prove. We recall that we had $\gamma' = \gamma\beta$, where β represents the slide of a hole along the leftmost useful arc b. We observe that we can construct a curve diagram of the braid β^{-1} such that the first curve b_1 of the diagram is a line segment in E_1 from -1 almost all the way to $E_1 \cap b$, followed by an arc parallel and close to the arc b, and finally running into the same hole as b. The construction of the arc b_1 is illustrated in figure 7(a).

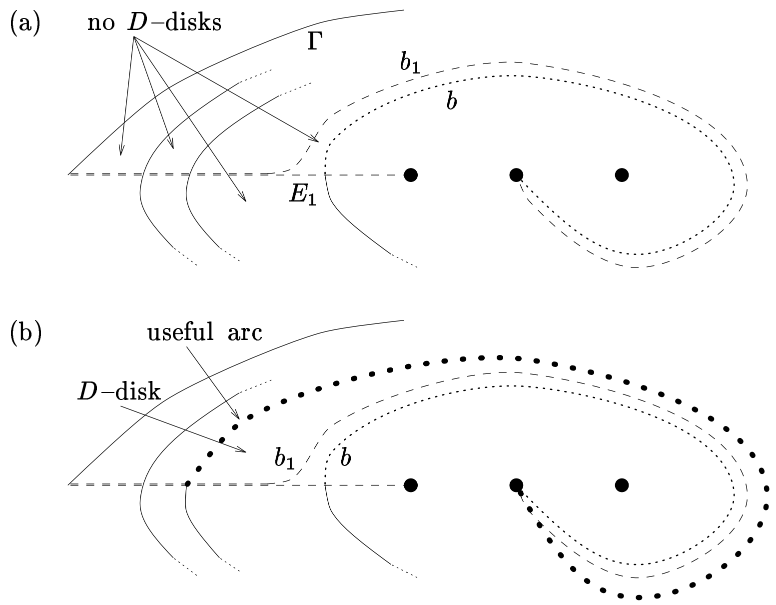

Figure 7: There are no D–disks between b_1 and Γ

Next we examine the possible reductions of Γ with respect to this arc b_1. If there was a D–disk of type (b) whose boundary contained the arc b, (ie to the right of b_1 in figure 7) then cutting off the strip bounded by b, b_1 and E_1 would yield a D–disk of type (a) of Γ with respect to E (see figure 7(a)). This is impossible by hypothesis. If there was a D–disk of type (b) whose boundary contained a final segment of the arc b_1 and a segment other than b of a curve of Γ, (ie to the left of b_1 in figure 7) then this segment would be a useful arc intersecting E_1 more to the left than b (figure 7(b)), which is also impossible. Finally, any D–disk of type (a) of Γ with respect to b_1 would also be a D–disk of Γ with respect to E_1. So there are no D–disks between b_1 and Γ. By remark 2.4 it follows that we can reduce the curve diagram of β^{-1} with respect to Γ without touching its first curve b_1. We can now observe that γ is 1–greater than β^{-1}, ie γ' is 1–positive, as claimed. This completes the proof of claim 2.

Finally we turn to the case when i may not be 1. In this case the first $i-1$ holes are lined up near -1 on the real axis. The same argument as in the case $i = 1$, only with the $i-1$st hole and the line segment E_i playing the role previously played by -1 and E_1 respectively, completes the proof of the general case. □

The proof of theorem 4.2 provides an explicit algorithm for converting a braid into its left-consistent canonical form. In the appendix we give a formal version of this algorithm using cutting sequences.

Remark The order on the braid group has the property that inserting a generator σ_i anywhere in a braid word makes the braid larger. A proof of this fact, in the spirit of this paper, is given in [17]. This property is equivalent to the statement that the order extends the *subword order* defined by Elrifai and Morton [8] and an algebraic proof has been given by Laver [11].

5 Counterexamples

We shall call a braid word *σ-consistent* (Dehornoy in [7] calls it *reduced*) if it is σ-positive, σ-negative or trivial. We have seen in the previous chapter that every braid has at least one σ-consistent representative. The aim of this chapter is to disprove some plausible-sounding but overoptimistic conjectures about the ordering and about σ-consistent representatives of braids.

Left invariance on the pure braid group

Because the *pure* braid group has an ordering which is simultaneously left and right invariant [15], it would be tempting to think that the geometric ordering is left and right invariant when restricted to the pure braid group. However this is equivalent to saying that a pure positive braid, when conjugated by any pure braid, is again positive and the example in figure 8 shows this to be false. In B_3, the braid group on three strings, we conjugate the pure positive braid $\sigma_1^2\sigma_2^{-2}$ by the pure braid $\sigma_2\,\sigma_1^2\,\sigma_2$. The figure shows the equivalence of the resulting braid with the σ–negative braid $\sigma_2^{-1}\sigma_1^{-1}\sigma_2^3\,\sigma_1^{-1}\sigma_2\,\sigma_1^{-1}$. We are moving first the string segment $\overline{ab}$ and then the segment $\overline{cd}$ 'over' the braid 'to the left of the braid'.

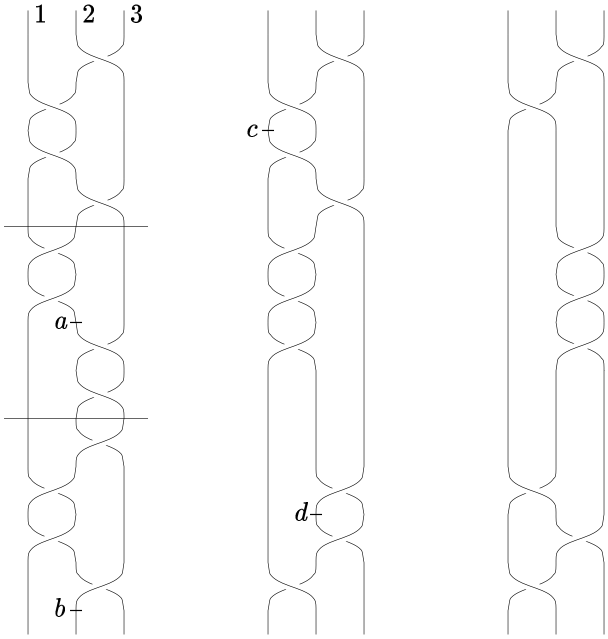

Figure 8: Equivalence of a conjugate of a positive pure braid with a visibly negative braid

Simultaneously shortest and σ-consistent representatives

For any element b of the braid group B_n $(n \geqslant 2)$, there are two ways to represent b by a particularly simple word w in the letters $\sigma_1^{\pm 1}, \ldots, \sigma_{n-1}^{\pm 1}$.

(1) b can be represented by a word which is as short as possible. For instance, we shall see later that the word $w_1 = \sigma_1\sigma_2\sigma_3^{-1}\sigma_2\sigma_1^{-1}$ is a shortest possible representative of a braid in B_4 (see figure 9(a)).

(2) b can be represented by a σ-consistent word. For instance, in the braid word $w_2 = \sigma_2^{-1}\sigma_3^{-1}\sigma_1\sigma_2^{-1}\sigma_1\sigma_3\sigma_2$, which represents the same element of B_4 as w_1, the letter σ_1 occurs only with positive exponent, see figure 9(b).

Theorem 5.1 *Every element of B_n for $n = 2, 3$ has a simultaneously shortest and σ-consistent representative. By contrast, there are braids in B_n for $n \geqslant 4$ all of whose σ-consistent representatives have non-minimal length.*

Proof The case $n = 2$ is obvious. The case $n = 3$ follows from the fact that in B_3 Dehornoy's handle-reduction algorithm [7] never increases the length of

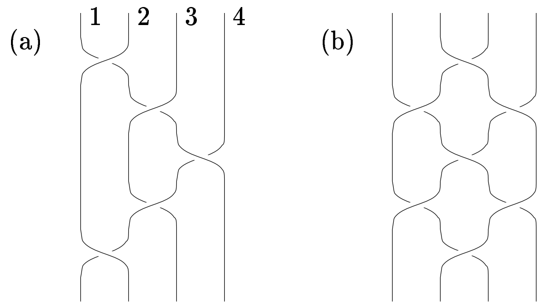

Figure 9: The equivalent braids $\sigma_1\sigma_2\sigma_3^{-1}\sigma_2\sigma_1^{-1}$ and $\sigma_2^{-1}\sigma_3^{-1}\sigma_1\sigma_2^{-1}\sigma_1\sigma_3\sigma_2$

a braid word, and hence turns any shortest representative of a given braid into a simultaneously shortest and σ-consistent one.

For the case $n \geqslant 4$ it suffices to prove that the braid $b := \sigma_1\sigma_2\sigma_3^{-1}\sigma_2\sigma_1^{-1} \in B_4$ (figure 9) has length 5, while every σ-consistent representative has more than five letters.

To see that every representative has at least five letters we note that the image of b under the natural homomorphism $B_n \to S_n$, from the braid group into the symmetric group, is the permutation (14). This permutation cannot be written as a product of less than five adjacent transpositions. The result follows.

We now assume, for a contradiction, that there exists a five-letter representative which is also σ-consistent. This would be a braid on four strands with the following properties:

(i) its image under the natural map $B_4 \to S_4$ is (14),

(ii) it has five crossings (ie it is a word with five letters),

(iii) if we denote by $c(i,j)$ $(i,j \in \{1,\dots,4\})$ the algebraic crossing number of the ith and the jth string, then the braid must satisfy $c(1,2) = 1$, $c(1,3) = 1$, $c(1,4) = -1$, $c(2,3) = 0$, $c(2,4) = -1$, $c(3,4) = 1$,

(iv) it may contain the letter σ_1, but not σ_1^{-1} (note that there exists a representative of b in which σ_1 occurs only positively, so there can't exist a consistently negative one).

There are only three braids satisfying (i) - (iii), pictured in figure 10, and we observe that none of them satisfies (iv). It follows that no σ-consistent representative of b with only five crossings exists. $\square$

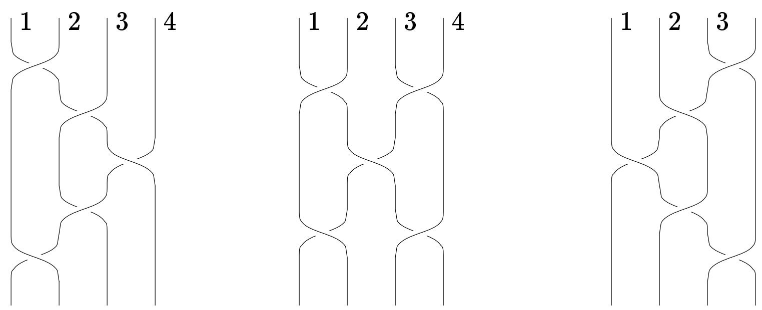

Figure 10: Three candidates for short σ-consistent representatives of b

Minimal number of occurrences of the main generator

We define the *main generator* of a braid word to be the generator with lowest index occurring in the word. It is tempting to think that sliding holes along *leftmost* useful arcs, as in the left consistent canonical form, is the most efficient way of reducing the number of intersections between the curve diagram and the line segment E_1. This, however, is wrong:

Theorem 5.2 *There are braids whose left consistent canonical form does not have the minimal number of occurrences of the main generator among all σ-consistent representatives.*

Proof We shall show that the braid Δ^3, where $\Delta = \sigma_2\sigma_1\sigma_2$, has this property. Note that Δ is just a half-twist, so Δ^2 generates the commutator subgroup of B_3.

We have $(\sigma_2\sigma_1\sigma_2)^3 = \sigma_2\sigma_2\sigma_1\sigma_2\sigma_2\sigma_2\sigma_1\sigma_2\sigma_2$, so the braid can be represented by a σ-consistent word in which the main generator σ_1 occurs only twice. However, as is easy to check with the help of figure 11, the left consistent canonical form of the braid is the word $(\sigma_2\sigma_1\sigma_2)^3$, which contains the main generator σ_1 three times. □

Local indicability

We are indebted to Stephen P Humphries and Jim Howie for pointing out the following. A group is called *locally indicable* if every finitely generated subgroup has a nontrivial homomorphism to the integers. It was proved by Burns and Hale [4] that locally indicable groups are right-orderable, but it took

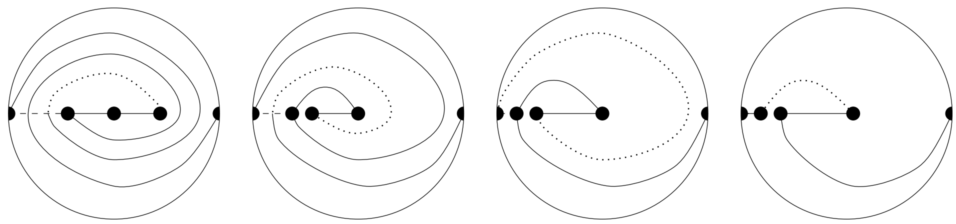

Figure 11: The left consistent canonical form of Δ^3 is $\sigma_2\sigma_1\sigma_2\sigma_2\sigma_1\sigma_2\sigma_2\sigma_1\sigma_2$

almost two decades until G Bergman [1] found an example of a group which is right-orderable but not locally indicable; ie the class of locally indicable groups is *strictly* contained in the class of right-orderable groups. We can now give further examples:

Theorem 5.3 *The braid group B_n for $n \geqslant 5$ is right orderable but not locally indicable.*

Proof It remains to show that B_n is not locally indicable. The commutator subgroup B_n' of B_n is finitely generated, and for $n \geqslant 5$ the first and second commutator subgroups coincide: $B_n' = B_n''$ (see [10]). It follows that the abelianization of B_n' is trivial, so $B_n' \subset B_n$ has no nontrivial homomorphism to $\mathbb{Z}$. □

6 Automatic ordering

Define a right-invariant ordering to be *automatic* if it can be determined by a finite-state automaton. In this section we shall see that the ordering on the braid group is automatic.

This is proved by comparing the order on the braid group as defined in section 3 with Mosher's automatic structure [12, 13]. This comparison gives more. Define a group to be *order automatic* if it is both automatic [9] and right-orderable and such that there is a finite state automaton which detects the order from the automatic normal forms. To be precise, there exists an automatic structure and a finite state automaton, which, given two normal forms for the automatic structure, will decide which represents the greater group element.

Theorem 6.1 *The braid group B_n is order automatic.*

Remark 6.2 The algorithm to decide which of two given normal forms is the greater takes linear time in the length of the normal form. Using results from Epstein et al [9] we deduce:

Corollary 6.3 *There is a quadratic-time algorithm to decide which of two elements of B_n (presented in terms of standard braid generators) is the greater.*

Full details of the proof of these results can be found in [16]. Here we shall give a short proof of theorem 6.1 which yields only a quadratic time algorithm to order normal forms which is nevertheless sufficient to imply corollary 6.3.

In [12, 13] Mosher constructs normal forms for elements of mapping class groups by *combing* triangulations (and hence proves that mapping class groups are automatic). We shall need to sketch Mosher's normal form in the special case of the braid group.

We define the *base triangulation* B of D_n to have vertices at the n missing points and at the four *boundary vertices*, ± 1 and $\pm\sqrt{-1}$. The edges of B comprise the four arcs of ∂D_n joining pairs of boundary vertices, $n+1$ edges along the real axis and $2n$ edges joining $\pm\sqrt{-1}$ to the real vertices not ± 1, see figure 12. We order and orient the edges as indicated.

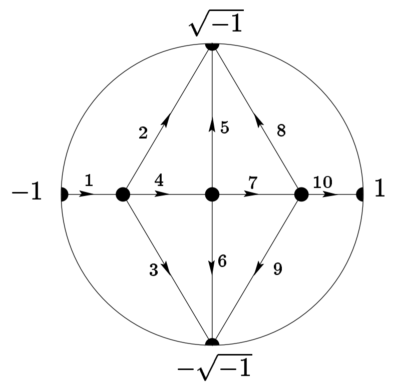

Figure 12: The base triangulation

An *allowable triangulation* of D_n is a triangulation with the same vertex set. We identify two allowable triangulations if they differ by a vertex fixing isotopy. A *triangulation class* is a set of boundary fixing isomorphism classes of allowable triangulations. Ie two triangulations are in the same class if they differ by an element of the braid group.

We now consider the groupoid $\mathcal{G}$ which has for objects the set of triangulation classes of D_n and for morphisms the set of ordered pairs (T, T') of allowable triangulations, where (T, T') is identified with $(h(T), h(T'))$ if $h \in B_n$. The morphism goes from the class of T to the class of T'. If T and T' are in the same class, then there is a unique boundary fixing isomorphism from T' to T up to isotopy, ie an element of B_n. This determines an isomorphism between the vertex group of $\mathcal{G}$ and the braid group B_n. (Note that for this isomorphism, and for compatibility with Mosher's conventions, we need to replace the *algebraic* convention for multiplication in the braid group, described in section 1, by the opposite *functional* convention, ie $\Phi\Psi := \Phi \circ \Psi$. The functional convention is used throughout this section; the algebraic convention is used in all other sections and in the appendix.)

Combing

We consider a particular type of morphism in $\mathcal{G}$.

Definition *Flipping an edge* An edge α adjacent to two triangles δ and δ' is removed (to form a square of which α is a diagonal) and then the square is cut back into two triangles by inserting the opposite diagonal. We call this morphism "flipping α" and denote it f_α, see figure 13.

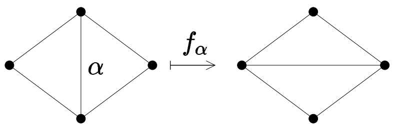

Figure 13: Flipping an edge

Every morphism $q = (B, T)$ in $\mathcal{G}$ from the base vertex to another vertex is a product of a canonical sequence of flips. To see this, picture q as given by superimposing B and T, and *comb* T along B. To be precise, first reduce T with respect to B and then consider edge 1 of B. Suppose that, starting at -1, edge one crosses edge α of T. Flip α. Repeat until there are no more crossings of edge 1 with T. (The fact that this process is finite follows from a simple counting argument: one counts the number of intersections of edge 1 with T, except with the next edge of T which is to be flipped. For more detail here see [13, pages 321–322].) Now do the same for edge 2 starting at the non-boundary vertex and continue in this way, using the ordering and orientation of edges of B indicated in figure 12, until T has been converted into a copy of B.

The *Mosher normal form* of q is the inverse of the sequence of flips described above.[2] Notice that unlike the general case described in [13], q is completely characterised by the sequence of flips, there is no need to carry the labelling of T along. particular, there is no relabelling morphism required here. (This is because ∂D_n is fixed throughout.)

Detecting order from the Mosher normal form

To see the connection with order, consider an element (B, T) of the vertex group at the class of B. There is an element $g \in B_n$ (a homeomorphism of D_n fixing ∂D_n) unique up to isotopy carrying T to B. Conversely given $g \in B_n$ the corresponding triangulation pair is $(B, g^{-1}B)$.

We observe that if we comb B along $g(B)$ this is combinatorially identical to combing $g^{-1}(B)$ along B. We call the sequence of flips defined by this combing the *combing sequence of* g. (The reverse of the combing sequence is the Mosher normal form of g.)

The curve diagram of g is part of the triangulation $g(B)$ namely the edges numbered $1, 4, 7, \ldots, 3n+1$. Suppose that g is i–positive, then g can be assumed to fix the first $i-1$ of these edges (ie $1, 4, \ldots, 3i-5$) and then, after reduction, can be assumed to fix the corresponding outlying edges (ie $2, 5, \ldots, 3i-4$ and $3, 6, \ldots, 3i-3$). But edge $3i-2$ is carried into the upper half of D_n and must meet edge $3i-1$ of B. Thus the first flip in the combing sequence of g is f_{3i-1}, ie flip the edge numbered $3i-1$. Similarly if g is i–negative then the first flip in the combing sequence is f_{3i}. We have proved the following:

Algorithm 6.4 (To decide from the Mosher normal form whether a braid element is i–positive or negative and provide the correct value of i)
Inspect the combing sequence (the reverse of the Mosher normal form). The first flip is either f_{3i-1} for some i or f_{3i} for some i. In the first case the braid is i–positive and in the second it is i–negative.

This algorithm is visibly executable by a finite-state automaton and linear in the length of the normal form of g. Theorem 6.1 and corollary 6.3 follow

[2] Strictly speaking the Mosher normal form is not this flip sequence, which only defines an asynchronous automatic structure, but is derived from it by clumping flips together into blocks called "Dehn twists", "partial Dehn twists" and "dead ends" (see [13] pages 342 et seq). This technicality does not affect any of the results proved here or in [16]. We prove that order can be detected in linear time from the flip sequence. Since the clumped flip sequence can be unclumped in linear time, this implies that order can be detected in linear time from the strict Mosher normal form.

from general principles. To decide the relative order of two elements α and β we compute the normal form of $\alpha\beta^{-1}$ — this can be done by a finite-state automaton and takes quadratic time, see [9] — and then apply algorithm 6.4.

Final remarks (1) We have proved that there is a quadratic time algorithm to decide the relative order of two braid words. In [7] Dehornoy presents an algorithm which does this in practice and is apparently extremely fast — however his formal proof that this algorithm works only provides an exponential bound on time. The algorithm presented here is implementable since the whole Mosher program can be implemented, see [14]. Note that in the appendix we present another algorithm based on cutting sequences.

(2) There is a far stronger connection between the Mosher normal form and the order on B_n than presented here. The relative order of two elements can be detected from their combing sequences by inspecting just the first four differences in the sequences (and this proves remark 6.2). Full details here are to be found in [16].

References

[1] **G M Bergman** *Right orderable groups that are not locally indicable*, Pacific J Math 174 (1991) 243–248

[2] **J Birman**, *Braids, links, and mapping class groups*, Annals of Math. Studies, 82, Princeton University Press, Princeton (1975)

[3] **J S Birman**, **C Series**, *An algorithm for simple curves on surfaces*, J. London Math. Soc (2) 29 (1984) 331–342

[4] **R G Burns, V W D Hale**, *A note on group rings of certain torsion free groups*, Canad Math Bull 15 (1972) 441–445

[5] **P Dehornoy**, *Braid groups and left distributive operations*, Trans. AMS 345 (1994) 115–150

[6] **P Dehornoy**, *From large cardinals to braids via distributive algebra*, J. Knot Theory and its Ramifications 4(1995) 33–79

[7] **P Dehornoy**, *A fast method of comparing braids*, Adv. in Math. 125 (1997) 200–235

[8] **E A Elrifai**, **H R Morton**, *Algorithms for positive braids*, Quart. J. Math. Oxford 45 (1994) 479–497

[9] **D B A Epstein et al**, *Word processing in groups*, Jones & Bartlett (1992)

[10] **E A Gorin, V Ja Lin** *Algebraic equations with continuous coefficients, and certain questions of the algebraic theory of braids*, Math USSR-Sb 7 (1969) 569-596

[11] **R Laver**, *Braid group actions on left-distibutive structures and well-orderings in the braid group*, J. Pure Appl. Algebra 108 (1996) 81–98

[12] **L Mosher**, *Mapping class groups are automatic*, Math. Research Letters 1 (1994) 249–255

[13] **L Mosher**, *Mapping class groups are automatic*, Annals of Math. 142 (1995) 303–384

[14] **L Mosher**, *A user's guide to the mapping class group: once punctured surfaces*, MSRI preprint

[15] **D Rolfsen**, **Jun Zhu**, *Braids, orderings and zero divisors*, submitted to J. Knot Theory and its Ramifications

[16] **C Rourke**, **B Wiest**, *Order automatic mapping class groups*, (to appear), `http://www.maths.warwick.ac.uk/~cpr/ftp/ordaut.ps`

[17] **B Wiest**, *Dehornoy's ordering of the braid groups extends the subword ordering*, Pacific J. Math. (to appear)

Addresses:

R. Fenn: *School of Mathematical Sciences, University of Sussex, Falmer, Brighton BN1 9QH, UK* `R.A.Fenn@sussex.ac.uk`

M. T. Greene: *Radan Computational, Ensleigh House, Granville Road, Bath BA1 9BE, UK* `Michael.Greene@uk.radan.com`

D. Rolfsen: *Department of Mathematics, University of British Columbia, Vancouver, B.C. Canada V6T 1Z2* `rolfsen@math.ubc.ca`

C. Rourke: *Mathematics Institute, University of Warwick, Coventry CV4 7AL, UK* `cpr@maths.warwick.ac.uk`

B. Wiest: *CMI, Université de Provence, 13453 Marseille cedex 13, France,* `bertw@gyptis.univ-mrs.fr`

A Appendix: Cutting sequences

In this appendix we define a unique *reduced cutting sequence* for a braid. We give implementable algorithms to read the reduced cutting sequence from the braid word, to decide order from the cutting sequence and to put a braid, given in terms of standard twist generators, into its left-consistent canonical form.

Cutting sequences and curve diagrams

A *cutting sequence* is a finite word χ in the letters $0, \ldots, n$, $\underline{0}, \ldots,$ $\underline{n+1}$, $\uparrow$ and $\downarrow$ such that

(i) χ starts with $\underline{0}$ and ends with $\underline{n+1}$,

(ii) each of the letters $\underline{0}, \ldots, \underline{n+1}$ occurs precisely once in χ,

(iii) in the word χ numbers and arrows alternate, with the single possible exception that strings of the form $\underline{i}\,\underline{i+1}$ or $\underline{i+1}\,\underline{i}$ $(i = 0, \ldots, n)$ may occur.

Consider now a curve diagram Γ. It consists of three types of subcurves: curves in the upper half plane, curves in the lower half plane, and straight line segments in the real line. Note that curves in the upper or lower half plane may be replaced by semicircles since they are determined by their end points. For convenience we rescale the curve diagram so that it goes from 0 to $n+1$ and the n holes are the integers $1, 2, \ldots, n$.

Going along Γ we can read off a cutting sequence, by reading an $\uparrow$ or $\downarrow$ for every curve in the upper or lower half plane respectively, an $\underline{i}$ $(i \in \{0, \ldots, n+1\})$ for every intersection with the integer i in the real line (so underlined integers correspond to holes), and an i for every intersection with the real interval $(i, i+1)$. It is easy to check that a word obtained in this way is indeed a cutting sequence.

For example the curve diagram representing σ_1 in figure 2 is coded as $\underline{0} \uparrow \underline{2}\,\underline{1} \downarrow \underline{3}\,\underline{4}$, whereas $\sigma_1\sigma_2^{-1}$ is coded $\underline{0} \uparrow 1 \downarrow \underline{3} \downarrow \underline{1} \downarrow 3 \uparrow \underline{2} \uparrow \underline{4}$.

We define a *reduction* of a cutting sequence to be a replacement of the sequence by a shorter one, according to the one of the following rules (where $\updownarrow$ denotes $\uparrow$ *or* $\downarrow$, and $i \in \{0, \ldots n\}$).

- $\underline{i} \updownarrow i \ \to\ \underline{i}$, $\underline{i+1} \updownarrow i \ \to\ \underline{i+1}$, $i \updownarrow \underline{i} \ \to\ \underline{i}$, $i \updownarrow \underline{i+1} \ \to\ \underline{i+1}$,
- $\downarrow i \downarrow \to \downarrow$, $\uparrow i \uparrow \to \uparrow$,
- $i \updownarrow i \ \to\ i$,
- $\underline{i} \updownarrow \underline{i+1} \ \to\ \underline{i}\,\underline{i+1}$, $\underline{i+1} \updownarrow \underline{i} \ \to\ \underline{i+1}\,\underline{i}$

A cutting sequence is called *reduced* if it allows no reduction.

Proposition A.1 *Every braid on n strings has a unique reduced cutting sequence.*

Proof Let χ be a cutting sequence of a curve diagram Γ of the braid. We observe that a reduced version χ' of χ is the same as the cutting sequence of a curve diagram Γ', where Γ' is obtained by reducing Γ with respect to the trivial curve diagram E. From proposition 2.3 we deduce that any two reduced cutting sequences χ' and χ'' must come from curve diagrams which are equivalent with respect to E. Therefore χ' and χ'' must agree. $\square$

The reduced curve diagram can be reconstructed from the reduced cutting sequence. Thus the cutting sequence classifies the curve diagram, and hence the braid. This is most easily seen by using pen and paper. One reads the cutting sequence, and for every number symbol one encounters, draws one arc in the diagram. If the cutting sequence is reduced, then this involves no choices. Below we shall give an algorithm to do this which is more suitable for computer implementation.

Note that it is easy to construct reduced cutting sequences which do not come from curve diagrams. The pen and paper method can also be used to decide whether a cutting sequence does correspond to a curve diagram. Again we give a more formal algorithm below which will do this.

Reading the cutting sequence from the braid word

We next show how to convert a braid defined in terms of the twist generators $\sigma_i^{\pm 1}$ into a reduced cutting sequence. We do this inductively by defining how σ_i and σ_i^{-1} act on reduced cutting sequences and then let the whole word act on the trivial sequence $\underline{0}\ \underline{1} \ldots \underline{n}\ \underline{n+1}$.

Algorithm A.2 *Suppose a braid β has reduced cutting sequence χ. Then a cutting sequence of $\beta\sigma_i$ is obtained by simultaneously making the following replacements everywhere in the word χ. These rules are to be interpreted as simultaneous, not sequential, replacements.*

(i) $\underline{i} \to \underline{i+1}, \quad \underline{i+1} \to \underline{i},$

(ii) $\downarrow (\underline{i}) \to \downarrow i-1 \uparrow (\underline{i+1}), \quad (\underline{i}) \downarrow \to (\underline{i+1}) \uparrow i-1 \downarrow,$

(iii) $\underline{i-1}\ (\underline{i}) \to \underline{i-1} \uparrow (\underline{i+1}), \quad (\underline{i})\ \underline{i-1} \to (\underline{i+1}) \uparrow \underline{i-1},$

(iv) $\uparrow (\underline{i}) \to \uparrow (\underline{i+1}), \quad (\underline{i}) \uparrow \to (\underline{i+1}) \uparrow,$

(v) $\downarrow (\underline{i+1}) \to \downarrow (\underline{i}), \quad (\underline{i+1}) \downarrow \to (\underline{i}) \downarrow,$

(vi) $\underline{i+2}\ (\underline{i+1}) \to \underline{i+2} \downarrow (\underline{i}), \quad (\underline{i+1})\ \underline{i+2} \to (\underline{i}) \downarrow \underline{i+2},$

(vii) $\uparrow (\underline{i+1}) \to \uparrow i+1 \downarrow (\underline{i}), \quad (\underline{i+1}) \uparrow \to (\underline{i}) \downarrow i+1 \uparrow.$

(viii) $\downarrow i \uparrow \to \downarrow i-1 \uparrow i \downarrow i+1 \uparrow, \quad \uparrow i \downarrow \to \uparrow i+1 \downarrow i \uparrow i-1 \downarrow,$

Note: in rules (ii) - (vii), *rule* (i) *is being applied, and its application is indicated by brackets. Replacements of symbols other than $\underline{i}, \underline{i+1}$ depend on context, eg rule* (ii) *says that if $\downarrow$ is followed by $\underline{i}$, then it is to be replaced by $\downarrow i-1 \uparrow$, and the $\underline{i}$ is replaced by $\underline{i+1}$, by* (i)*. So $\downarrow \underline{i}$ turns into $\downarrow i-1 \uparrow \underline{i+1}$.*

The rules for the action of σ_i^{-1} are obtained by interchanging the symbols $\uparrow$ and $\downarrow$ everywhere in this list (ie replacing up- by down-, and down- by up-arrows). The resulting cutting sequence can then be reduced, to obtain the reduced cutting sequence of the braid $\beta\sigma_i$ or $\beta\sigma_i^{-1}$.

We can now deduce an effective algorithm to decide whether a given braid is positive, trivial, or negative:

Algorithm A.3 (To decide if a given braid is positive, trivial, or negative)
Use algorithm A.2 to calculate the reduced cutting sequence of the braid. The braid is positive if and only if the first arrow in this sequence is an up-arrow $\uparrow$.

Recovering the curve diagram from the cutting sequence

We now show how to recover a reduced curve diagram from its associated cutting sequence. At the same time this will provide an effective algorithm to decide if a given cutting sequence corresponds to a curve diagram.

To make precise the problem here, we define the *real cutting sequence* of a curve diagram to be the cutting sequence, with the non-underlined integers replaced by real numbers specifying the precise intersection point of the curve diagram with the real line, up to order preserving bijections. (Taking the integer part of all numbers in the real cutting sequence we retrieve the cutting sequence.) Given the real cutting sequence, we can immediately construct the curve diagram. Moreover it is trivial to check if a real cutting sequence corresponds to an (embedded) curve diagram: one just checks that

(1) if $\underline{i}\ \underline{i+1}$ or $\underline{i+1}\ \underline{i}$ occurs in the sequence then no real number in $(i, i+1)$ occurs,

(2) the numbers on each side of two arrows of the same type correspond to nested intervals (so that the corresponding curves do not intersect).

So we need an algorithm to reconstruct the real cutting sequence from the cutting sequence or equivalently to decide for each i the order in which the corresponding points actually occur in $\mathbb{R}$.

Algorithm A.4 *Suppose the letter* i $(i \in \{0, \ldots, n\})$ *appears in two different places, say in the* r*th and* s*th position, in the cutting sequence. To decide which one represents the smaller number in the interval* $(i, i+1)$ *in the real cutting sequence proceed as follows.*

Since the cutting sequence is reduced, there are two arrows in opposite direction adjacent to each of the letters i. Starting at the rth letter we read the sequence either forwards or backwards. We define the *up-string at the* r*th place* to be the word obtained from the cutting sequence by reading forwards or backwards, starting at the rth letter, up to the next underlined number, with the reading direction specified by the requirement that the the first two letters read should be $i \uparrow$. Similarly, we define the *down-string at the* r*th place* by reading in the opposite direction, such that the resulting word starts with $i \downarrow$, again up to the next underlined number. We compare the up-string at the rth with that at the sth place, and the down-string at the rth with that at the sth place. They cannot both agree, for if they did, the curve diagram would have two curves with the same endpoints.

We now manipulate the up- and down strings as follows: firstly, we increase all non-underlined integers by $\frac{1}{2}$. Then we remove the underline from all underlined integers. We obtain sequences of the form $x_0 \updownarrow x_1 \updownarrow \ldots \updownarrow x_{l-1} \updownarrow x_l$, where $l \in \mathbb{N}$, $x_0 = i + \frac{1}{2}$, $x_1, \ldots, x_{l-1} \in \{\frac{1}{2}, 1\frac{1}{2}, \ldots, n + \frac{1}{2}\}$, and $x_l \in \{0, \ldots n+1\}$.

From this we can construct a sequence of numbers in $\{1, 1\frac{1}{2}, \ldots, n - \frac{1}{2}, n\}$, called the *cyclically associated sequence*, as follows. For every string $x_j \uparrow x_{j+1}$ we write down the unique representative in $\{\frac{1}{2}, 1, \ldots, n, n+\frac{1}{2}\}$ of $x_{j+1} - x_j + (n+1)\mathbb{Z} \in \mathbb{R}/(n+1)\mathbb{Z}$; for every string $x_j \downarrow x_{j+1}$ we write down the unique representative in $\{\frac{1}{2}, 1, \ldots, n, n+\frac{1}{2}\}$ of $x_j - x_{j+1} + (n+1)\mathbb{Z} \in \mathbb{R}/(n+1)\mathbb{Z}$. Altogether, this yields a sequence of length l.

We now define an up-string u to be *cyclically lexicographically larger* than another up-string u', if the cyclically associated sequence of u is lexicographically larger than the one of u'.[3] The geometric interpretation is that the curve diagram has two line segments starting in the real interval $(i, i+1)$, going into the upper half plane. The line segment representing the cyclically lexicographically larger up-string is the one turning 'more to the left'. Since the two line segments must be disjoint (being part of the curve diagram), the starting point of the curve segment yielding the cyclically lexicographically larger up-string must represent a smaller real number in the real cutting sequence. Similarly, we define a cyclic lexicographic ordering on the down-strings; this time, the starting point of a curve segment which gives rise to a cyclically lexicographically larger down-string than another curve segment must represent a *larger* real number in the real cutting sequence. *End of algorithm A.4*

To summarise, we have found an algorithm for reconstructing the real cutting sequence from the cutting sequence: given any two places in the cutting sequence where the letter i occurs, we compare the up-strings at these places. If they agree, we compare the down-strings instead. In either case we can work out the cyclically associated sequences, and then decide which of the two letters i represents the smaller number in the interval $(i, i+1)$ in the real cutting sequence.

An algorithm to determine order from the cutting sequence

Algorithm A.4 also allows us to decide which of two given reduced cutting sequences represents the larger braid. If the two sequences agree on some initial segment, then we remove the underlines from all underlined numbers (except the first letter $\underline{0}$) that lie in this segment. Then we reduce the resulting two sequences. We obtain two new sequences whose initial segments up to the first underlined numbers do not agree. If they differ already on the second letter (after $\underline{0}$), then we know which one is larger. Otherwise, we work out which of them is cyclically lexicographically larger, using algorithm A.4.

[3] Cyclic lexicographic order is used by Birman and Series [3].

The algorithm to determine left-consistent canonical form

We are finally ready to describe our algorithm to calculate the left-consistent canonical form of a braid. The input is a braid β represented as a word w in the twist generators $\sigma_i^{\pm 1}$. The output is the same braid in left-consistent canonical form of β, again given as a word in the $\sigma_i^{\pm 1}$.

The algorithm proceeds by repeating the main step (described below) after each repetition we have a word W and a cyclically reduced cutting sequence χ which are both modified at the next repetition.

Start We start with W the trivial word, and χ the reduced cutting sequence of β calculated using algorithm A.2.

Finish If the reduced cutting sequence χ is $\underline{0}\,\underline{1}\ \ldots \underline{n}\ \underline{n+1}$, then the algorithm stops, and the inverse of the word W is the desired canonical word.

Main step If the reduced cutting sequence starts $\underline{0}\ \underline{1}\ldots\underline{i}\uparrow$, with $i < n+1$, then we hunt for subwords of the following forms

(i) $i \uparrow a_1 \downarrow a_2 \uparrow \ldots \updownarrow a_{l-1} \updownarrow \underline{a_l}$ or
(ii) $\underline{i} \uparrow a_1 \downarrow a_2 \uparrow \ldots \updownarrow a_{l-1} \updownarrow \underline{a_l}$ or
(iii) $\underline{a_l} \updownarrow a_{l-1} \updownarrow \ldots \uparrow a_2 \downarrow a_1 \uparrow i$ or
(iv) $\underline{a_l} \updownarrow a_{l-1} \updownarrow \ldots \uparrow a_2 \downarrow a_1 \uparrow \underline{i}$,

where the $a_1, \ldots, a_{l-1}$ are not equal to i and not underlined, and $a_l \neq i, i+1$. (If the reduced cutting sequence starts $\underline{0}\ldots\underline{i}\downarrow$, then we hunt for subwords like $i \downarrow a_1 \uparrow \ldots \updownarrow a_{l-1} \updownarrow a_l$ instead.) We shall call these words *useful subwords*, because they correspond to useful arcs.

We consider the set of all useful subwords, and we want to identify the 'leftmost one', ie the one whose letter i represents the leftmost point in the interval $(i, i+1)$. If one of them starts or ends with a letter $\underline{i}$, ie if one of them is of type (ii) or (iv), then this is it. If not, then we can use algorithm A.4 to determine the leftmost one. When we have found the leftmost useful subword, we modify it as follows. If it is of type (i) or (ii), then we write it backwards, so that it starts with the letter $\underline{a_l}$. Irrespectively of the type of the useful subword, we remove the underline from the letter $\underline{a_l}$. Then we let $c := a_l$, replace all letters a_k $(k \in \{1, \ldots, l\})$ with $a_k \geqslant c$ by $a_k - 1$ (eg a_l turns into $a_l - 1$), and reduce the resulting sequence. By doing this, we obtain a modified sequence $a'_0 \updownarrow a'_1 \updownarrow \ldots \updownarrow a'_{l'}$, possibly with the letter $a'_{l'} = i$ underlined.

We now multiply W on the right by a word $v_1 \ldots v_l$, where v_k is determined by a'_{k-1}, a'_k, and the arrow in between a'_{k-1} and a'_k as follows:

(i) If the modified leftmost useful subword contains the string $a'_{k-1} \uparrow a'_k$, and $a'_{k-1} < a'_k$, then $v_k = \sigma_{a'_{k-1}+1} \cdots \sigma_{a'_k}$;

(ii) If the modified leftmost useful subword contains the string $a'_{k-1} \uparrow a'_k$, and $a'_{k-1} > a'_k$, then $v_k = \sigma^{-1}_{a'_{k-1}} \dots \sigma^{-1}_{a'_k+1}$;

(iii) If the modified leftmost useful subword contains the string $a'_{k-1} \downarrow a'_k$, and $a'_{k-1} < a'_k$, then $v_k = \sigma^{-1}_{a'_{k-1}+1} \dots \sigma^{-1}_{a'_k}$;

(iv) If the modified leftmost useful subword contains the string $a'_{k-1} \downarrow a'_k$, and $a'_{k-1} > a'_k$, then $v_k = \sigma_{a'_{k-1}} \dots \sigma_{a'_k+1}$

The word $v_1 \dots v_l$ represents the slide of a hole back along the leftmost useful arc.

Finally, we calculate the new reduced cutting sequence after this slide. This can be done by letting the word $v_1 \dots v_l$ act on the reduced cutting sequence, as described above. (An alternative method would be to remove the underline from the letter $\underline{a_l}$, underline the unique letter i which belongs to the leftmost useful subword instead, carefully relabel the cutting sequence, using algorithm A.4, and then reduce the resulting cutting sequence.) *End of main step*

The proof of theorem 4.2 implies that the algorithm stops after a finite number of repetitions of the main step.