Algorithms for positive braids

ELSAYED A. ELRIFAI [1]
Department of Mathematics, Faculty of Science,
King Khalid University, P.O.Box 9004, Abha, Saudi Arabia.
aaalrefaai@kku.edu.sa

HUGH R. MORTON
Department of Mathematical Sciences, University of Liverpool,
Peach St, Liverpool, L69 7ZL, England.
morton@liv.ac.uk

Abstract

We give an easily handled algorithm for the word problem in each of Artin's braid groups, B_n, based on Garside's methods, but framed more directly in terms of the set of positive braids in which each pair of strings crosses at most once.

We develop a natural partial order on each braid group defined in terms of positive braids, and apply this to compare braids with different powers Δ^r of the fundamental half-twist braid Δ. This leads to an improvement of Garside's conjugacy algorithm, using a much smaller finite subset of each conjugacy class, which we term the super summit set, to represent the class, in place of Garside's summit set.

Introduction

The emphasis in this paper is on the set of positive braids, those elements of Artin's braid group B_n which can be written as a word in positive powers of the usual generators, $\sigma_1, \ldots, \sigma_{n-1}$. We shall regard elements of B_n as geometric braids, given up to isotopy by an arrangement of n strings running monotonically from top to bottom between two parallel discs. We study the braid group in terms of the set S_n^+ of positive permutation braids. These are defined to be positive braids in which each pair of strings crosses at most once, and they are shown to be in 1-1 correspondence with the permutations in S_n. Two canonical forms for positive braids as products of braids in S_n^+ are given, a left and a right handed form, along with a readily mechanised algorithm for writing every braid in canonical form. This is a useful technique

[1]The first author was supported during this work by a scholarship from the Egyptian Ministry of Education.

for handling braids, as permutations can be dealt with very easily, and the algorithm is capable of quick hand use in diagrammatic form for short braids.

The algorithm and related developments provide us with:

1. An easily handled approach to Garside's solution of the word problem in B_n.

2. An improvement of Garside's solution of the conjugacy problem, by reducing his study of the 'summit set' to that of a much smaller invariant class under conjugation, the 'super summit set'.

3. An algorithm to decide whether $(\Delta_n)^k$ is a factor of a given positive braid, where the fundamental braid Δ_n is the positive braid in B_n in which each pair of strings crosses exactly once. This happens if and only if at least k of the canonical factors of the braid are equal to Δ_n.

4. An algorithm to decide whether a given positive braid is a factor of $(\Delta_n)^k$. This happens if and only if its canonical form has at most k factors.

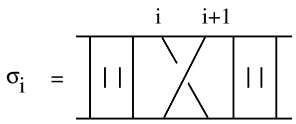
In the next section properties (3) and (4) are recast in the framework of a natural partial order on B_n. Some features of B_n which become obvious in this setting serve to clarify the organisation of (1) and (2).

Since a general braid can always be expressed as the product of a positive braid and a, possibly negative, power of Δ_n our study extends readily to the general case. The algorithms as given here were originally described in [5]. The groundwork and key lemmas originate from Garside [6] via Birman [2], while one very useful result from our point of view comes from the appendix to Garside's thesis [7]. The presentation and emphasis has been considerably adapted here, and results in a much shorter conjugacy algorithm. With very little change a more even-handed view can be given, using both positive and negative braids with no special preference for positive; such an approach has been recently described by Thurston [10]. We note in section 5 the minor modifications needed to pass from one view to the other for a general non-positive braid.

The class of links which arise as the closure of positive braids is of some independent geometric interest. All such links are fibred [9], and they include as special cases the algebraic links, arising from isolated singularities of polynomials in $\mathbf{C}^2$, and Lorenz links, which appear as orbits of a certain dynamical system [3, 4]. The algorithms presented here can be used to tackle questions about special classes such as the Lorenz links [5].

1 Artin's braid group

Artin's classical result [1] gives a presentation for the group B_n of braids on n strings by generators $\sigma_1, \ldots, \sigma_{n-1}$, where σ_i represents the geometric braid in which strings i and $i+1$ cross once only as indicated below.



The relations in his presentation are

1. $\sigma_i \sigma_j = \sigma_j \sigma_i, \quad |i-j| > 1,$
2. $\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}, \quad 1 \leq i \leq n-2.$

Definition. A *positive* braid is an element of B_n which can be written as a word in positive powers of the generators $\{\sigma_i\}$, without the use of the inverse elements σ_i^{-1}.

A *positive braid word* is an explicitly written word $w(\sigma_1, \ldots, \sigma_{n-1})$.

Notation. Write B_n^+ for the set of positive braids in B_n.

Remark. The abelianisation map defines a natural homomorphism wt : $B_n \to \mathbf{Z}$, by $\mathrm{wt}(\sigma_i) = 1$, which coincides with the length of a positive braid word on B_n^+.

There is an automorphism $\tau : B_n \to B_n$, which can be thought of as 'turning over' a braid. It is defined by $\tau(\sigma_i) = \sigma_{n-i}$.

One other map suggested by symmetries of geometric braids is the 'reversing' *anti* automorphism rev : $B_n \to B_n$, defined by $\mathrm{rev}(\sigma_i) = (\sigma_i)$. This has the effect on a geometric braid of turning it over from top to bottom, when it will be read as the reverse word in the braid generators.

Notation. Write $\Delta_n \in B_n$ for the 'fundamental braid', described as a geometric n-braid by imagining the strings attached to a rod which is given a positive half-twist. It can be defined inductively by

$$\Delta_n = \Delta_{n-1} \sigma_{n-1} \sigma_{n-2} \cdots \sigma_1$$

starting with $\Delta_1 = \sigma_1$.

Although Δ_n is not immediately recognisable in B_n from a group-theoretic point of view, its square generates the centre of B_n, while it is readily shown that the automorphism τ is conjugation by Δ_n.

Before giving details of our algorithms and their proofs we shall make some further definitions to put our extension of Garside's conjugacy algorithm into a natural context. We shall make considerable use of comparisons between our given braid element and powers of Δ_n; our algorithms will allow us to make these comparisons very explicitly.

We start by using the positive braids B_n^+ to define a partial order on B_n.

Notation. For $A, B \in B_n$ write $A \leq B$ when $B = C_1 A C_2$ for some $C_1, C_2 \in B_n^+$.

We then have

$$B \in B_n^+ \Leftrightarrow e \leq B$$

and

$$A \leq B \Leftrightarrow B^{-1} \leq A^{-1}.$$

In what follows we shall write Δ for Δ_n unless there is any danger of ambiguity.

Lemma 1.1 *Each generator σ_i satisfies $e \leq \sigma_i \leq \Delta$.*

Proof: Immediate from the inductive definition of Δ. □

Lemma 1.2 *If $A \leq \Delta^s$ then $\Delta^s = D_1 A = A D_2$ for some $D_1, D_2 \in B_n^+$.*

Proof: Write $\Delta^s = C_1 A C_2$ with $C_1, C_2 \in B_n^+$. Then

$$\Delta^s = \tau^s(C_2) C_1 A = A C_2 \tau^s(C_1),$$

since $\Delta^s C_2 = \tau^s(C_2)\Delta^s$. □

Similarly,

Lemma 1.3 *If $\Delta^r \leq A$ then $A = E_1 \Delta^r = \Delta^r E_2$ for some $E_1, E_2 \in B_n^+$.*

Garside's original algorithms make considerable use of 'initial braids' and 'final braids', which can be expressed in our terminology as braids B with $e \leq B \leq \Delta$, using lemma 1.2. We shall show later how these braids, which we call *positive permutation braids,* can be readily described geometrically, and identified with permutations in S_n, which we exploit as a ready means of referring to them without having to use explicit braid words.

Lemmas 1.2 and 1.3 give the immediate corollary:

Corollary 1.4 *If $\Delta^{r_1} \leq B \leq \Delta^{s_1}$ and $\Delta^{r_2} \leq C \leq \Delta^{s_2}$ then*

$$\Delta^{r_1+r_2} \leq BC \leq \Delta^{s_1+s_2}.$$

We may then place every braid somewhere between powers of Δ.

Theorem 1.5 *Every B satisfies $\Delta^r \leq B \leq \Delta^s$ for some $r, s \in \mathbf{Z}$.*

Proof: Write B as a word in $\{\sigma_i^{\pm 1}\}$ and use corollary 1.4, and the fact that $e \leq \sigma_i \leq \Delta$ and $\Delta^{-1} \leq \sigma_i^{-1} \leq e$. □

Notation. Write $[r, s] \subset B_n$ for the subset $\{B \in B_n : \Delta^r \leq B \leq \Delta^s\}$.

We shall use the common group-theoretic notation for subsets of groups, in which $ST = \{st \in G : s \in S, t \in T\} \subset G$ where S, T are subsets of a group G. Then $[r, s] = \Delta^r[0, s - r]$, and we may study a braid by looking at the 'braid intervals' in which it lies.

Clearly there will be a shortest braid interval $[r, s]$ containing a given braid B.

Definition. For $B \in B_n$ set $\inf B = \max\{r : \Delta^r \leq B\}$ and $\sup B = \min\{s : B \leq \Delta^s\}$. Call $\ell(B) = \sup B - \inf B$ the *canonical length* of B.

It is a consequence of corollary 1.4 that $\ell(AB) \leq \ell(A) + \ell(B)$.

Remark. In Garside's terminology, the *power* of B is exactly the same as $\inf B$, since we have $r = \inf B$ if and only if $B = \Delta^r B'$ with $e \leq B'$ and $\Delta \not\leq B'$. This last condition is referred to by Garside as 'B' is coprime to Δ'.

Although he did not introduce a counterpart of $\sup B$, it may be noted that $-\sup B = \inf B^{-1}$ is the power of B^{-1} in Garside's sense.

While the analogy with real intervals is useful, it is not close enough to behave well under unions. However, corollary 1.4 can be read as

$$[r_1, s_1][r_2, s_2] \subset [r_1 + r_2, s_1 + s_2],$$

and we shall show later that this inclusion is an equality. Consequently any B can be written as the product of Δ^r with $\ell(B)$ braids in $[0, 1]$, where $r = \inf B$. This factorisation, which can be chosen uniquely under a further hypothesis, forms the basis of the algorithm for the word problem, and is essentially that given by Garside.

The conjugacy problem was solved by Garside by noting that there are only finitely many braids B' of the same weight wt B' with $\Delta^r \leq B'$, so that there is a maximum value of $\inf B'$ among braids B' which are conjugate to a given braid B, called the 'summit power' of B. There are then finitely many conjugates of B having this summit power, and these constitute the 'summit set' of B. Since two elements are conjugate if and only if they have the same summit sets, the conjugacy algorithm is completed by an algorithm to construct the summit set. This is done by showing that there is a chain of conjugates leading from B to any element of its summit set in which the power never decreases along the chain, and successive elements are conjugate by some element of the finite set of braids $[0,1]$.

We show that the same result applies to the 'super summit set' of a braid B, which is the subset of the summit set consisting of braids B' of minimal canonical length $\ell(B')$. The resulting algorithm, in constructing a smaller, more constrained set, is then quicker and more practical.

2 Positive braids

We can study a braid $B \geq \Delta^r$ by studying the positive braid $\Delta^{-r}B$ so this section will concentrate on the set B_n^+ of positive braids. Garside works initially with positive words regarded as elements of the semigroup generated by $\sigma_1, \ldots, \sigma_{n-1}$, with the Artin relations. This imposes an equivalence relation, which he denotes by $\underset{\cdot}{=}$, on the set of positive words. While equivalent words are equal in B_n it is conceivable that the converse does not hold. Garside proves an embedding theorem, to show that words which are equal in B_n are also equal in the semigroup.

We shall make use of a key lemma of Garside, which can be proved readily for the semigroup, but relies on his embedding theorem for its application to B_n.

Notation. Write

$$\sigma_i * \sigma_j = \begin{cases} \sigma_i, & i = j \\ \sigma_i \sigma_j, & |i-j| > 1, \\ \sigma_i \sigma_j \sigma_i, & |i-j| = 1. \end{cases}$$

Lemma 2.1 (Garside). *Let $P = \sigma_i P_1 = \sigma_j P_2$ with $P_1, P_2 \geq e$. Then $P_3 \geq e$, where $P = (\sigma_i * \sigma_j) P_3$.*

We shall look at factorisations of positive braids into positive factors, drawing on this lemma to guarantee the uniqueness of our factorisation under suitable conditions.

Definition. The *starting set* , $S(P) \subset \{1, \ldots, n-1\}$, for a positive braid $P \geq e$, is the set

$$S(P) = \{i : P = \sigma_i P_i,\ P_i \geq e\}.$$

Similarly the *finishing set* of P is

$$F(P) = \{i : P = P_i \sigma_i,\ P_i \geq e\}.$$

Clearly $F(P) = S(\operatorname{rev} P)$.

Definition. A positive factorisation $P = AB$ with $A, B \geq e$ is a *left-weighted* factorisation if $S(B) \subset F(A)$, and is *right-weighted* if $S(B) \supset F(A)$.

We want to factorise a given positive braid successively by left-weighted factorisations $P = A_1P_1, P_1 = A_2P_2, \ldots$ where the left-hand factor A_i lies in $[0, 1]$ in each case. Such a factorisation is the heart of Garside's solution to the word problem.

Lemma 2.2 *Every $P \geq e$ has a unique left-weighted factorisation $P = A_1P_1$ with $A_1 \in [0, 1]$. Every other positive factorisation $P = AB$, with $A \in [0, 1]$, satisfies $A_1 = AQ$ for some $Q \geq e$.*

Before giving the proof we discuss the braids in $[0, 1]$ from a more geometric viewpoint, so that their properties may be more easily recognised, and the braids handled more readily.

For a positive geometric braid we can count the number of crossings, which will always be in the same sense, between any chosen pair of strings. This number is not altered by isotopy of the braid, and will not depend on which explicit positive braid word is used to represent the braid.

Definition. A braid $A \geq e$ is called a *positive permutation braid* if it can be drawn as a geometric braid in which every pair of strings crosses at most once.

Notation. Write S_n^+ for the set of positive permutation braids.

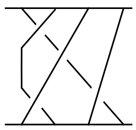
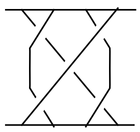
We shall shortly show that S_n^+ consists exactly of the set $[0, 1]$. First we show how S_n^+ corresponds bijectively to the set S_n of permutations.

Lemma 2.3 *If the braids $A_1, A_2 \in S_n^+$ induce the same permutation on their strings then $A_1 = A_2$. For each $\pi \in S_n$ there is a braid $A_\pi \in S_n^+$ which induces that permutation.*

Proof: Number the strings of each braid $1, \ldots, n$ according to the top point of each string. Then strings i and j have at most one crossing point, where string j passes in front of string i if $i < j$. Each braid can then be drawn in a box in which string 1 lies in a vertical plane at the furthest back level, with the other strings in a succession of vertical planes lying further forward. Because A_1 and A_2 induce the same permutation on their strings the ith string in each braid runs to the same point at the bottom of the braid, and one braid can be moved to the other by isotopy keeping each string in its vertical plane.

To construct A_π it is enough to find a geometric braid with permutation π in which pairs of strings cross at most once. Arrange n points at the top and bottom of a rectangle, and draw lines in the plane joining point i at the top to point $\pi(i)$ at the bottom, so that pairs of lines cross at most once, and only two lines cross at any one point. This is a familiar diagrammatic way of visualising the permutation π. Convert this into a braid diagram by separating the lines at every crossing to make a positive crossing. This requires that the line from i crosses under the line from j if $i < j$. □

Examples of two such braids, P_1 with permutation (143) and P_2 with permutation (14), are shown below. They can be written as explicit braid words in several ways, for example $P_1 = \sigma_1\sigma_2\sigma_3\sigma_1$ and $P_2 = \sigma_1\sigma_3\sigma_2\sigma_3\sigma_1$. The great benefit of this lemma is however that it is quite unnecessary to remember these braids as braid words; the permutation is quite enough, and makes for much greater ease in comparing such braids.

P_1  P_2 

It can be seen from a diagram, for example, exactly what the starting set of a braid in S_n^+ must be.

Lemma 2.4 *For $A_\pi \in S_n^+$ the following are equivalent:*

1. *$i \in S(A_\pi)$,*
2. *strings i and $i+1$ cross in A_π,*
3. *$\pi(i+1) < \pi(i)$.*

Proof: Clearly (2) and (3) are equivalent since strings cross at most once. If $A_\pi = \sigma_i A'$ with $A' \geq e$ then (2) follows, since the strings cross in σ_i. If (2) holds then a diagram of the permutation can be drawn in which this crossing is the first to take place. Construction of A_π as a braid from this diagram gives a braid word starting with σ_i. □

We can then see immediately that the two braids P_1 and P_2 in the figure above have $S(P_1) = \{1,2\}$, $F(P_1) = \{1,3\}$, $S(P_2) = F(P_2) = \{1,3\}$, using lemma 2.4 applied to each braid and its reverse.

Lemma 2.5 *Let $A \in S_n^+$. Then $\sigma_i A \in S_n^+$ if and only if $i \notin S(A)$.*

Proof: Strings i and $i+1$ in $\sigma_i A$ cross once if $i \notin S(A)$ and twice otherwise, while all other pairs cross at most once. $\square$

Remark. An extension of this proof shows that if $i, j \notin S(A)$ then $\sigma_i * \sigma_j A \in S_n^+$, while application of proposition 2.5 to $\operatorname{rev} A$ shows that similar results hold for the finishing set. Calculations of the starting sets for A_π can be made very quickly using the factorial coordinates described in [8] when listing the permutations $\pi \in S_n$.

Theorem 2.6 *The subsets* $[0,1]$ *and* S_n^+ *of* B_n *are identical.*

Proof: Clearly $\Delta \in S_n^+$ as every pair of strings crosses once. It is also immediate that if $P = AB \in S_n^+$ with $A, B \geq e$ then any pair of strings in A can cross at most once, since they have at most one crossing in the whole braid P. Thus A, and equally B will also lie in S_n^+. Now any $A \in [0,1]$ satisfies $AB = \Delta$ for some $B \geq e$, so $A \in S_n^+$.

Conversely, suppose that $A = A_\pi \in S_n^+$. Let $\delta \in S_n$ be the permutation of Δ, namely $\delta(i) = n - i$. Let ρ be the permutation with $\pi\rho = \delta$. Then $A_\pi A_\rho$ is a positive braid with permutation $\pi\rho = \delta$ so it is enough to show that it lies in S_n^+ to deduce that $A_\pi A_\rho = A_\delta = \Delta$ and hence that $A_\pi \in [0,1]$. Now any pair of strings in $A_\pi A_\rho$ can cross at most twice. Since the resulting permutation is δ each pair of strings crosses an odd number of times, and hence each pair crosses exactly once. $\square$

The braid Δ satisfies $S(\Delta) = F(\Delta) = \{1, \ldots, n-1\}$, since we have shown in 1.1 that $\sigma_i \leq \Delta$ for each i. The converse holds, as shown in the next lemma.

Lemma 2.7 *Let* $A \in S_n^+$ *satisfy* $S(A) = \{1, \ldots, n-1\}$. *Then* $A = \Delta$.

Proof: Let A have permutation π. Then $\pi(i) > \pi(i+1)$ for each i, since $i \in S(A)$. Thus $\pi(i) > \pi(j)$ for each $i < j$, so that strings i and j cross in A for each i, j. Then A is the positive permutation braid in which every pair of strings crosses, so $A = \Delta$. $\square$

The same result follows with $F(A)$ in place of $S(A)$, by applying lemma 2.7 to $\operatorname{rev} A$.

We now return to the proof of lemma 2.2, using the identification of the sets $[0,1]$ and S_n^+, and the properties of S_n^+ which we have just established.

Proof: Proof of lemma 2.2 We start by showing the existence of a left-weighted factorisation $P = A_1 P_1$ with $A_1 \in S_n^+$.

Consider all positive factorisations $P = AB$ with $A \in S_n^+$, and select one in which $\operatorname{wt} A$ is maximal. If $S(B) \not\subset F(A)$ then we can find $i \in S(B)$

with $i \notin F(A)$. By lemma 2.5 we have $A' = A\sigma_i \in S_n^+$ and $B = \sigma_i B'$ with $B' \geq e$ giving another positive factorisation $P = A'B'$ with $A' \in S_n^+$ and $\mathrm{wt}\, A' > \mathrm{wt}\, A$, so the selected factorisation must be left-weighted. Write this factorisation as $P = A_1 P_1$.

We now show that every other positive factorisation $P = AB$ with $A \in S_n^+$ is a subfactorisation, in the sense that $A_1 = AQ$ for some $Q \geq e$. Otherwise there exist factorisations $P = C\sigma_i B'$ with $C\sigma_i \in S_n^+$ such that $C \in S_n^+$ is a subfactor of A_1 but $C\sigma_i$ is not. Choose such a factorisation with largest possible $\mathrm{wt}\, C$, and write $A_1 = CQ$. Now $\mathrm{wt}\, A_1 \geq \mathrm{wt}\, A\sigma_i > \mathrm{wt}\, A$, by the maximality of $\mathrm{wt}\, A_1$, so $Q \neq e$. We may then choose $j \in S(Q)$. Then $C\sigma_j \leq A_1$ and so $C\sigma_j \in S_n^+$. Write the resulting factorisation as $P = C\sigma_j B''$. Apply Garside's lemma to $B = \sigma_i B' = \sigma_j B''$ to see that $P = C(\sigma_i * \sigma_j)B'''$ for some $B''' \geq e$.

Now $C(\sigma_i * \sigma_j) \in S_n^+$, by the remark following lemma 2.5. This yields a factorisation of P with a larger subfactor (at least containing $C\sigma_j$) in common with A_1 while $C(\sigma_i * \sigma_j)$ is not itself a subfactor of A_1.

It follows immediately that the left-weighted factorisation $P = A_1 P_1$ is unique, for if $P = AB$ is another such, then we can write $A_1 = AQ$ with $Q \geq e$. Either $Q = e$ and $A = A_1$ as claimed, or we can find $i \in S(Q)$. Then $i \notin F(A)$, since $A\sigma_i \leq A_1 \in S_n^+$. However $B = QP_1$, so $i \in S(B)$ and the factorisation $P = AB$ is not left-weighted. $\square$

Corollary 2.8 *Let $P \geq e$ have left-weighted factorisation $P = A_1 P_1$, with $A_1 \in S_n^+$. Then $S(A_1) = S(P)$.*

Proof: Clearly $S(A_1) \subset S(P)$.

Let $i \in S(P)$. Then $P = \sigma_i B$ with $B \geq e$, so by lemma 2.2 we have $A_1 = \sigma_i Q$ for $Q \geq e$, and hence $i \in S(A_1)$. $\square$

As a result we have the *left-canonical form* for positive braids as follows:

Theorem 2.9 *There is a unique expression for $P \geq e$ as $P = A_1 A_2 \cdots A_k$ with $A_i \in [0, 1]$, $A_k \neq e$ and $S(A_{i+1}) \subset F(A_i)$ for each i.*

Proof: Take $P_i = A_{i+1} \cdots A_k$, and $P_0 = P$. Then $A_i P_i$ is the unique left-weighted factorisation of P_{i-1}. This follows by downward induction on i since then $S(P_i) = S(A_{i+1})$ from corollary 2.8. $\square$

Remark. We can find $\inf P$ immediately from this expression for P, by observing that $P \geq \Delta$ if and only if $A_1 = \Delta$. For if $P \geq \Delta$ then $P = \Delta Q$ for $Q \geq e$. Then $S(P) = \{1, \ldots, n-1\}$ and so, by corollary 2.8, $S(A_1) = \{1, \ldots, n-1\}$. Then lemma 2.7 ensures that $A_1 = \Delta$. Using lemma 2.7 again we can see that if $A_i = \Delta$ in the left-canonical form, then $A_j = \Delta$ for all $j < i$, so that

$$\inf P = \max\{i : A_i = \Delta\}.$$

An extension of this result will be used in identifying $\sup P$.

Lemma 2.10 *Let $P \geq e$ have left-weighted factorisation $P = A_1 P_1$, with $A_1 \in [0, 1]$. If $B \geq e$ and $BP \geq \Delta$ then $BA_1 \geq \Delta$.*

Proof: By induction on $\mathrm{wt}\, B$. The result holds for $B = e$. Otherwise we can write $B = B'\sigma_i$ and $P' = \sigma_i P$ to get $B'A'_1 \geq \Delta$, where $A'_1 \in [0, 1]$ is the initial left-weighted factor of $P' = A'_1 P'_1$. Now $i \in S(P')$ so $i \in S(A'_1)$, by corollary 2.8. Write $A'_1 = \sigma_i A''_1$. Then $A''_1 \in [0, 1]$, and $P = A''_1 P'_1$. From proposition 2.2 we have $A_1 = A''_1 Q$ for some $Q \geq e$. Now $BA''_1 = B'A'_1 \geq \Delta$, and hence $BA_1 = BA''_1 Q \geq \Delta$. □

Theorem 2.11 *Let $P \geq e$ have left-canonical form $P = A_1 A_2 \cdots A_k$. Then $\sup P = k$.*

Proof: By induction on k.

Write $s = \sup P$. Then $s \leq k$, since $P \in [0, k]$ by corollary 1.4. Now $s \geq 1$, unless $P = e$ when there is nothing to prove. We can then find $B \geq e$ with $BP = \Delta^s$. Now $BA_1 \geq \Delta$ by lemma 2.10, so that $BA_1 = \Delta B_1$ with $B_1 \geq e$. Then $\Delta^s = BP = \Delta B_1 P_1$, where $P_1 = A_2 \cdots A_k$, so $B_1 P_1 = \Delta^{s-1}$, and $P_1 \leq \Delta^{s-1}$. By induction, $\sup P_1 = k - 1$, giving $k - 1 \leq s - 1$ and thus $s = k$. □

For a general braid $P \geq \Delta^r$ we may write $P' = \Delta^{-r} P \geq e$ and then $\sup P = r + \sup P'$ which can be calculated from the canonical form of the positive braid P', as can $\inf P = r + \inf P'$.

Having identified $\inf P$ and $\sup P$ in terms of the left-canonical form of a suitable positive braid we now give an explicit algorithm, based on the factorisation criterion in theorem 2.9, which will put a positive braid in left-canonical form, and so implement Garside's solution of the word problem.

Remark. We can also define the right-canonical form for a positive P, by writing $P = A_1 \cdots A_k$, with $A_i \in [0, 1]$, $A_1 \neq e$ and $S(A_{i+1}) \supset F(A_i)$ for each i, based on successive right-weighted factorisations of P. The right-canonical factorisation is exactly the reverse (all factors and their order reversed) of the left-canonical factorisation of $\mathrm{rev}\, P$. It follows immediately that $\inf P$ and $\sup P$ can be calculated in a similar way from the right-canonical form, and in particular, the number of factors will be the same in each case.

A similar analysis can be based on $S_n^- = [-1, 0]$, the set of negative permutation braids, to give negative canonical forms. The left-canonical form of a positive braid P based on S_n^+ yields the right-canonical form of P^{-1} based on S_n^-.

3 The word algorithm

The aim is to start from a word P and write $P = \Delta^r P'$ where $P' \geq e$ but $P' \not\geq \Delta$ and give the left-canonical form of P' as a sequence of permutations. Any other word $Q = P$ will give the same r and sequence of permutations, by the uniqueness in theorem 2.9.

Algorithm

Suppose then that we have written P in some way as $\Delta^r P'$, and P' is the product $B_1 B_2 \cdots B_k$ of positive permutation braids. Find the sets $F(B_i)$ and $S(B_i)$. If $S(B_{i+1}) \subset F(B_i)$ for each i then, by theorem 2.9, we have reached the left-canonical form for P', except possibly for some final factors of e. Incorporate any initial factors of Δ in the initial power of Δ; the remaining terms give the left-canonical form for P. The output of the algorithm is the power of Δ and the sequence of permutations defining the permutation braids.

Otherwise find the first i for which $S(B_{i+1}) \not\subset F(B_i)$ and select $j \in S(B_{i+1})$ with $j \notin F(B_i)$. Then $C_i = B_i \sigma_j$ and $C_{i+1} = \sigma_j^{-1} B_{i+1}$ both lie in S_n^+. Use them to replace B_i, B_{i+1} in the factorisation, and continue as before. This completes the algorithm.

Proof: The replacement gives a higher weighted sequence, in dictionary order of weights, and there are a finite number of sequences of a fixed total weight, so the process terminates. $\square$

To start the algorithm we must write the braid as the product of a power of Δ, to take account of negative letters in the given braid word, and a positive braid word. This can certainly be done by rewriting each σ_i^{-1} as $\Delta^{-1}\sigma_i^*$ with $\sigma_i^* \in [0, 1]$ and then collecting all the powers of Δ to the left, although it is generally possible to handle consecutive negative letters more efficiently. It remains to express the positive braid word as a product of positive permutation braids. There is nothing to do, other than to observe that each σ_i lies in $[0, 1]$, but a more efficient start can be made by reading the braid word from the left, and taking the longest initial subword that lies in $[0, 1]$ as the first permutation braid, before continuing.

Implementation

For an algebraic implementation the most useful technique is to have a list of the elements of S_n with a table of products of each $\pi \in S_n$ with the elementary transpositions $\tau_i = (i\ i+1)$, on the left and on the right, and a means of checking whether $\pi(i) > \pi(i+1)$. Given also the reverse map $\mathrm{rev} : S_n \to S_n$ for our listing we can then find immediately the sets $S(A_\pi)$ and $F(A_\pi)$ for each positive permutation braid. The test for the starting and finishing sets on adjacent permutations in the sequence can then be carried out quickly. If the two permutation braids have to be adjusted by moving σ_j from the right to the left then the new permutations can be found readily from the left and right multiplication information in S_n.

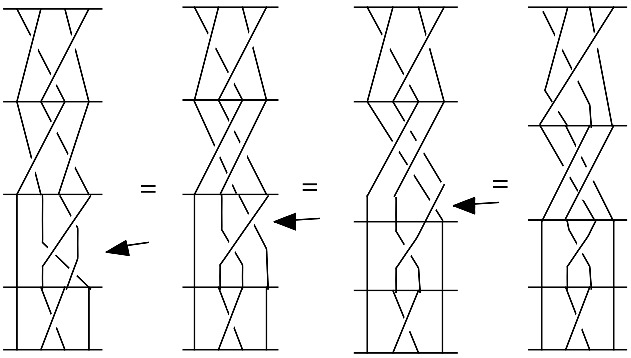
Practical storage of the information can be arranged by following the 'factorial coordinate' method, [8], for listing permutations $g = 1, \ldots, n!$ in terms of an expression $g = 1 + g_1 1! + g_2 2! + \ldots + g_{n-1}(n-1)!$ with $0 \leq g_i \leq i$. The factorial coordinates g_i count the number of crossings of string $i+1$ with lower numbered strings in the permutation g. When this ordering for the permutations is used, a table for left multiplication of the permutation g by τ_j can be readily constructed, and the product permutation is greater than g if and only if the braid $\sigma_j A_g$ is again a positive permutation braid. This provides a ready check on the set $S(A_g)$.

The storage that is needed can then be limited to the left multiplication table, an array of $n! \times (n-1)$ integers, and the reverse map, listing the permutations $\mathrm{rev}\, g$ for each g as a further array of $n!$ integers. These allow for right multiplication and checking of $F(A_g)$ by using left multiplication on the reverse braids.

A geometric example

Where a positive braid is given geometrically, it can be surprisingly easy to use the visual approach suggested by lemma 2.4 to arrive at its left-canonical form.

Let $P = \sigma_1\sigma_3\sigma_2^2\sigma_3\sigma_1\sigma_3\sigma_2\sigma_3\sigma_2$. Working down this braid as in the figure below we can partition it into permutation braids by continuing until a pair of strings are about to cross for the second time, and then starting a new permutation braid. Then look at adjacent pairs of permutation braids, and see if any adjacent pair of strings crosses in the lower but not the upper braid. If so, move it up and continue, otherwise stop.



The result in our example is a sequence of moves finishing with

$$P = (\sigma_1\sigma_3\sigma_2\sigma_1)(\sigma_2\sigma_1\sigma_3\sigma_2)(\sigma_2)(\sigma_2)$$

from which we have $\inf P = 0$ and $\sup P = 4$ together with a description of the left-canonical form by a sequence of four permutations.

4 The conjugacy algorithm

As outlined in section 2, this depends on listing all the conjugates of the given P in the interval $[r, s]$ where r is as large as possible, and s is as small as possible, given r. We have to show that a finite process of conjugating with braids in $[0, 1]$ will yield the complete list. The following 'convexity theorem' provides the backing for the algorithm, and is an adaptation of a result of Garside.

Theorem 4.1 *Let $P, Q \geq \Delta^r$ be conjugate. We may suppose that $A^{-1}PA = Q$ for some $A \geq e$. Then $A_1^{-1}PA_1 \geq \Delta^r$ where $A_1 \in [0, 1]$ is the first factor in the left-canonical form of A.*

Proof: The requirement that $A \geq e$ can be made without loss of generality, for if $PB = BQ$ we can write $B = \Delta^{2j}A$ with $A \geq e$ and then $PA = AQ$ also.

It is enough to prove the theorem in the cases $r = 0$ and r=1, for we may consider $P' = \Delta^{-2j}P$ and $Q' = \Delta^{-2j}Q$, when $A^{-1}P'A = Q'$.
Case 1 Given that $A^{-1}PA = Q$ and $P, Q \geq e$ we must show that $P_1 = A_1^{-1}PA_1 \geq e$. Now $\Delta A_1^{-1} = A_1^* \in S_n^+$, so

$$A_1^*PA = A_1^*AQ = A_1^*A_1A'Q = \Delta A'Q,$$

where $A = A_1A'$. Then $A_1^*PA \geq \Delta$, and we can apply lemma 2.10 with A_1^*P in the role of B and A in the role of P to get $A_1^*PA_1 \geq \Delta$, and hence $P_1 = \Delta^{-1}A_1^*PA_1 \geq e$.

Case 2 Now suppose that $A^{-1}PA = Q$ with $P, Q \geq \Delta$. Write $P = P'\Delta, Q = Q'\Delta$, and factorise $A = A_1A'$ as before. Then $\tau(A) = \Delta^{-1}A\Delta = \tau(A_1)\tau(A')$, giving a left-weighted factorisation of $\tau(A)$.

As above, $A_1^*PA = A_1^*\Delta A'Q$ so

$$A_1^*P'\tau(A)\Delta = \Delta A'Q'\Delta$$

and thus $A_1^*P'\tau(A) \geq \Delta$. Apply 2.10 again to get $A_1^*P'\tau(A_1) \geq \Delta$. Now

$$\begin{aligned}
\tau(P_1) &= \tau(A_1^{-1})\tau(P)\tau(A_1) \\
&= \Delta^{-1}\tau(A_1^*)\Delta P'\tau(A_1) \\
&= A_1^*P'\tau(A_1) \\
&\geq \Delta.
\end{aligned}$$

Hence $P_1 \geq \Delta$ also. □

Corollary 4.2 *Let $P, Q \in [r, s]$ be conjugate. Then there is a sequence $P = P_0, P_1, \ldots, P_k = Q$ of braids, all in $[r, s]$, such that each element is conjugate to the next by an element of $[0, 1]$.*

Proof: Take A with $A^{-1}PA = Q$ and write A in left-canonical form as $A = A_1 \cdots A_k$. Set $P_i = A_i^{-1}P_{i-1}A_i$. The result follows by induction on k once we show that $P_1 \in [r, s]$. Now $P_1 \geq \Delta^r$ by theorem 4.1, so it remains to show that $P_1 \leq \Delta^s$.

Now $P^{-1}, Q^{-1} \geq \Delta^{-s}$ and $A^{-1}P^{-1}A = Q^{-1}$ so by theorem 4.1 again we have $P_1^{-1} = A_1^{-1}P^{-1}A_1 \geq \Delta^{-s}$ and thus $P_1 \leq \Delta^s$. □

The general algorithm then is a procedure to construct the super summit set of the given braid P.

Definition. The *super summit set* of a braid P is the set of conjugates P' of P with $\inf P'$ maximal, and $\sup P'$ minimal within these.

Although it is conceivable that no conjugate simultaneously achieves a minimum for sup and a maximum for inf, we shall see from a subsidiary algorithm that $\sup P'$ is in fact an overall minimum on the super summit set, so that it could equally be defined as the set of conjugates of P which have minimal canonical length.

Algorithm

The conjugacy problem can now be solved for P and Q by determining the super summit set for P, finding one element of the super summit set for Q and comparing them. All elements of the super summit set of P can be found by conjugating repeatedly with elements of $[0, 1]$, while discarding any elements for which inf decreases or sup increases. Where $P \in [r, s]$ there are only finitely many conjugates of P within the interval, and these will all be found by the algorithm, by corollary 4.2. In fact the required interval may well be contained strictly in $[r, s]$; if one conjugate in a subinterval is found, then all previous conjugates with wider bounds may be discarded in the search, which stops when no conjugates of elements on the list by elements of $[0, 1]$ produce any new elements in the interval.

The next lemma allows us to give a quick algorithm to find the 'summit power' of P, i.e. the maximum value of inf on its conjugacy class. It relies on 'cycling' the left-canonical form of P to give a conjugate, as defined below. If any conjugate of P has a higher power then one such will be found by repeatedly cycling P, as shown in the next lemma.

Definition. Let $P = \Delta^r P_1 P_2 \cdots P_k$ where $r = \inf P$ and $P_1 \cdots P_k$ is the left-canonical form of the positive braid $\Delta^{-r}P$. Then $P_1 \neq \Delta$ and we can form the conjugate $c(P) = \Delta^r P_2 \cdots P_k \tau^r(P_1)$ which we say is given by *cycling* P.

Notice that $c(P)$ may not be given in left-canonical form directly, but application of the word algorithm shows that $\inf c(P) \geq r = \inf P$ and $\sup c(P) \leq k + r = \sup P$.

In fact $\inf c(P) \leq \inf P + 1$ and $\sup c(P) \geq \sup P - 1$ since if $P = A^{-1}QA$ or AQA^{-1} with $A \in [0, 1]$ and $Q \in [r', s']$ then $P \in [r' - 1, s' + 1]$ by corollary 1.4.

Lemma 4.3 *Suppose that P is conjugate to Q with $\inf Q > \inf P$. Then repeated cycling will produce $c^j(P)$ with $\inf c^j(P) > \inf P$, for some j.*

Proof: Let $Q = APA^{-1}$ with $A \geq e$, and let $\inf Q > r = \inf P$. The proof is by induction on $\mathrm{wt}\, A$.

We have $A\Delta^r P' = QA$, where $P = \Delta^r P'$ and $Q = \Delta^r Q'$ with $Q' \geq \Delta$. Then $\tau^r(A)P' = Q'A \geq \Delta$, so by 2.10 $\tau^r(A)P_1 \geq \Delta$, where $P_1 \neq \Delta$ is the first term in the left-canonical form of P'. Now $A\tau^r(P_1) \geq \Delta$, and we can write $A\tau^r(P_1) = A'\Delta$ where $A' \geq e$. Since $\tau^r(P_1) \leq \Delta$ we can write $\Delta = A''\tau^r(P_1)$ with $A'' \geq e$. Then $A = A'A''$ with $\mathrm{wt}\, A' = \mathrm{wt}\, A - \mathrm{wt}\, A'' < \mathrm{wt}\, A$ by the assumption that $P_1 \neq \Delta$. Now $\tau^r(P_1)c(P) = P\tau^r(P_1)$ by definition of $c(P)$, and so

$$\begin{aligned} QA\tau^r(P_1) &= AP\tau^r(P_1) \\ &= A\tau^r(P_1)c(P). \end{aligned}$$

Since $A\tau^r(P_1) = A'\Delta$ this gives $QA'\Delta = A'\Delta c(P)$. On extracting the factor of Δ we get $\tau(Q)\tau(A') = \tau(A')c(P)$ so that $c(P)$ is conjugate to $\tau(Q)$ by the positive braid $\tau(A')$ with smaller weight than A. Now $\inf \tau(Q) = \inf Q$, so either $\inf c(P) > \inf P$ or, by induction on the weight, some further cycling of $c(P)$ will lead to an increased value of inf. □

Corollary 4.4 *In every conjugacy class the maximum value of* inf *and the minimum value of* sup *can be achieved simultaneously. Thus the super summit set for a braid is the subset of its conjugacy class on which the canonical length ℓ is minimum.*

Proof: Let P achieve the minimum value of sup and let Q be a conjugate with the maximum value of inf. If $\inf P < \inf Q$ then repeated cycling of P will increase inf without increasing sup until both extreme values are achieved.□

Thus the process of repeated cycling of P will lead to the maximum value of inf, recognised when cycling starts to repeat conjugates already found. Notice that cycling a second time requires explicit calculation of the canonical form for $c(P)$ and cannot be done immediately from the form for P. Thurston [10] claims that if $\inf P$ is not the maximum value, then inf increases immediately, at the *first* cycling, $c(P)$. This would give a very quick test indeed to find the summit power, but unfortunately it is not true, as the following example, due to Birman, shows.

Example. Let $P = \sigma_1\sigma_2^2\sigma_3\sigma_1\sigma_2^2$. Then the left canonical form can be readily found from the word algorithm to be $P = (\sigma_1\sigma_2)(\sigma_2\sigma_3\sigma_1\sigma_2)(\sigma_2)$, giving $\inf P = 0$ and $\sup P = 3$. We have

$$\begin{aligned} c(P) &= (\sigma_2\sigma_3\sigma_1\sigma_2)(\sigma_2)(\sigma_1\sigma_2) \\ &= (\sigma_2\sigma_3\sigma_1\sigma_2)(\sigma_2\sigma_1\sigma_2) \\ &= (\sigma_2\sigma_3\sigma_1\sigma_2\sigma_1)(\sigma_2\sigma_1). \end{aligned}$$

This is now in canonical form, and still $\inf c(P) = 0$, although $\sup c(P) = 2$. However one further cycling gives $c^2(P) = (\sigma_2\sigma_1)(\sigma_2\sigma_3\sigma_1\sigma_2\sigma_1) = \Delta\sigma_2$, so that $\inf c^2(P) = 1$ and $\sup c^2(P) = 2$. Thus the change in value of inf is not realised after the first cycling.

The value of inf will not be altered by further cycling, which will continue to give $\Delta\sigma_2$.

In general, the extreme values for inf and sup on the conjugacy class of P can be found by cycling both P and P^{-1} until no new conjugates arise, recalling that $\sup P = -\inf P^{-1}$.

It is helpful to note that cycling P^{-1} for this purpose can be replaced by 'reverse cycling' P, where the last factor in the left-canonical form is moved to the beginning. Explicitly, let $P = \Delta^r P_1 P_2 \cdots P_k$ where $r = \inf P$ and $P_1 \cdots P_k$ is the left-canonical form of the positive braid $\Delta^{-r}P$, as above, and define $r(P) = \Delta^r \tau^r(P_k) P_1 \cdots P_{k-1}$.

Lemma 4.5 *We have* $(r(P))^{-1} = \tau(c(P^{-1}))$*.*

Proof: We can write
$$P^{-1} = \Delta^{-r-k} P'_k \cdots P'_1,$$
where
$$\tau^{r+i-1}(P'_i) P_i = \Delta = P_i \tau^{r+i}(P'_i).$$
This is actually the left canonical form for P^{-1}. It is enough to check that $S(P'_i) \subset F(P_{i+1})$ for each i. Now if $AB = \Delta$ with $A, B \geq e$ we have $F(A) \cup S(B) = \{1, \ldots, n-1\}$ and $F(A) \cap S(B) = \phi$, since every pair of strings in Δ crosses once and once only. Then
$$\begin{aligned} F(P_i) \cup S(\tau^{r+i}(P'_i)) &= \{1, \ldots, n-1\} \\ F(\tau^{r+i}(P'_{i+1})) \cup S(P_{i+1}) &= \{1, \ldots, n-1\} \end{aligned}$$
while $F(P_i) \cap S(\tau^{r+i}(P'_i)) = \phi = F(\tau^{r+i}(P'_{i+1})) \cap S(P_{i+1})$.

Given that P is in left canonical form, we have $S(P_{i+1}) \subset F(P_i)$. It follows that $S(\tau^{r+i}(P'_i)) \subset F(\tau^{r+i}(P'_{i+1}))$, and so $S(P'_i) \subset F(P'_{i+1})$.

Now we have $c(P^{-1}) = \Delta^{-r-k} P'_{k-1} \cdots P'_1 \tau^{r+k}(P'_k)$, and we can then confirm, by multiplication, that $r(P)\tau(c(P^{-1})) = e$. □

Then $\sup r(P) = -\inf \tau(c(P^{-1})) = -\inf c(P^{-1})$, and so $\sup r^j(P) = -\inf c^j(P^{-1})$. Since we can find the largest value of inf on conjugates of P^{-1} by considering only $c^j(P^{-1})$ we will thus find the smallest value of sup on all conjugates of P by considering only the successive reverse cycles $r^j(P)$. We can thus identify at least one element of the super summit set of P by cycling and reverse cycling, until repetition occurs, without calculating the whole super summit set.

In applying the algorithm to compare two elements P and Q we need only use cycling and reverse cycling on Q, having found the whole super summit set for P. Unfortunately, cycling is not enough in general to generate the whole super summit set, for example when $P = \sigma_1$ the other conjugates σ_i in the super summit set will not appear by cycling. Restriction to consideration only of the super summit set, coupled with the listing of elements by means of permutations does, however, give a much more effective algorithm than Garside's original. It should be practical to handle braids up to 6 strings at least by this method, and we hope to make a computer implementation shortly.

5 Concluding remarks

We have shown in corollary 1.4 that $[r_1, s_1][r_2, s_2] \subset [r_1 + r_2, s_1 + s_2]$.

Lemma 5.1 *We have* $[r_1, s_1][r_2, s_2] = [r_1 + r_2, s_1 + s_2]$.

Proof: It is enough to prove when $r_1 = r_2 = 0$. Let $P \in [0, s_1 + s_2]$ and let P have left canonical form $P = A_1 A_2 \cdots A_k$. By theorem 2.11 $k = \sup P \leq s_1 + s_2$, so we can factorise $P = P'P''$ with $P' = A_1 \cdots A_{s_1} \in [0, s_1]$ and $P'' \in [0, s_2]$. $\square$

A special example of this occurs where P is a general braid, neither positive or negative. Then $P \in [r, s]$ with $r \leq 0, s \geq 0$ and can be written as a product $P = P'P''$ of a negative and a positive braid, with $P' \in [r, 0]$ and $P'' \in [0, s]$. Set $k = s - r > -r > 0$, and suppose that $r = \inf P, s = \sup P$. Then $P = \Delta^r P_1 P_2 \cdots P_k$, using the left-canonical form, and $P' = \Delta^r P_1 \cdots P_{-r} \leq e$ while $P'' = P_{1-r} \cdots P_k$. Now $P' = Q_1 Q_2 \cdots Q_{-r}$ where $Q_i = \tau^{i-r}(\Delta^{-1} P_i)$ and $Q_i \in [-1, 0]$ is a negative permutation braid. The left-weighting condition for the original decomposition of P ensures that the representation of P' is the right-canonical form based on negative permutation braids, and also at the interface between P' and P'' we have $F(P') \cap S(P'') = \phi$.

Thurston notes this decomposition as a unique factorisation of P into a product of a negative and a positive braid, in which no cancellation into shorter braids can take place, for conversely any such factorisation, when written in terms of the canonical forms for the two halves, can then reproduce directly the left-canonical form for P. Similarly a unique factorisation as a positive times a negative braid arises from the right-canonical form of P.

We noted in theorem 2.6 that braids in $[0, 1]$ had a nice geometric characterisation in terms of string crossings. There is no corresponding generalisation of this to $[0, s]$. Any $P \in [0, s]$ is a factor of Δ^s in the sense that $PQ = \Delta^s$ for some $Q \geq e$. It follows that each pair of strings in P cross at most s times. This is not, however, a sufficient condition for P to lie in $[0, s]$, when $s > 1$.

For example, neither $\sigma_1^2 \sigma_2^2$ or $\sigma_2^2 \sigma_1 \sigma_3 \sigma_2^2$ lies in $[0, 2]$ although each pair of strings crosses no more than twice. A different combinatorial approach by

Casson shows however that if $P \geq e$ and every pair of strings crosses *exactly* twice then $P = \Delta^2$; this is the upper limit, as there are positive braids other than Δ^3 with exactly three crossings per pair of strings.

It was the attempt to explore the nature of the factors of Δ^s, in other words the sets $[0, s]$, from this geometric point of view of string crossings which led us to the formulation based on positive permutation braids, and eventually, through studying the details of Garside's thesis, to the algorithms presented here. By repeated applications of lemma 5.1 we can see that the positive factors of Δ^s are exactly the products of s positive permutation braids.

Version 2.3 June 1991, slightly amended from 2.2 of July 1990 (version 1, June 1988). Converted to Latex June 2004.

References

[1] Artin, E. Theory of braids. *Ann. Math.* 48 (1947), 101-126.

[2] Birman, J.S. Braids, links and mapping-class groups . *Annals of Maths. Studies* 82 (1974), Princeton University Press.

[3] Birman, J.S. and Williams, R.F. Knotted periodic orbits in dynamical systems I: Lorenz equations. *Topology* 22 (1983), 47-82.

[4] Birman, J.S. and Williams, R.F. Knotted periodic orbits in dynamical systems II *Contemporary Mathematics* 20 (1983), 1-60.

[5] Elrifai, E.A. Positive braids and Lorenz links. PhD Thesis, Liverpool University, 1988.

[6] Garside, F.A. The braid group and other groups . *Quart. J. Math. Oxford* (2) 78 (1969), 235-254.

[7] Garside, F.A. The theory of knots and associated problems . D.Phil. Thesis, Oxford University, 1965.

[8] Morton, H.R. and Short, H.B. Calculating the 2-variable polynomials of knots. *Journal of Algorithms* 11 (1990), 117-131.

[9] Stallings, J. Constructions of fibred knots and links . *Proc. Sympos. Pure Math. 32 part 2* (1978), 55-59.

[10] Thurston, W. Finite state algorithms for the braid group. Preprint 1988.