Symmetry of a Pastry
A Mathematical Investigation into the Braid Group

Ethan Partida

August 7, 2020

Abstract

In this paper we introduce braid groups from two perspectives, physical and algebraic. While a physical interpretation provides good intuition for the braid group, it lacks the precision that is useful for proving properties of this group. By first giving an introduction to the algebraic structure of the symmetric group and then shifting this perspective to the more generalized braid group, we gain the precision our physical interpretation lacks. Finally, we use this understanding to explore applications of the braid group in cryptography and knot theory.

Contents

1 Introduction

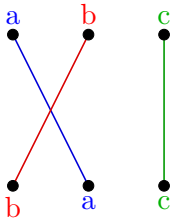
I recently participated in a reading course centered around Björner and Brenti's Coxeter groups text[1]. Throughout the reading, we identified things that looked like "$s_1s_2s_1 = s_2s_1s_2$" as braid relations. This naming had no concrete meaning to me. My best guess was that H.S.M Coxeter was just a particularly peckish mathematician, see Figure 1. Not surprisingly, it turns out that this was not the case. Its name comes from a rich theoretical background which somehow encompasses topology, quantum computing, knot theory and pastries. This paper will be a short dive into some of this background.

Figure 1: A *braid* relation, yum!

2 The Symmetric Group

To understand the theory behind braid relations we will first detour and learn about the symmetric group. The symmetric group, S_n, is the group of permutations of a set of size n. A permutation is a rearrangement of objects, e.g. "In what ways can I rearrange all six the books on my shelf?" To illustrate this, consider S_3. It contains all the permutations of the set $\{a, b, c\}$. As an example, $(\mathbf{12})$ is an element of this group. It is the permutation which swaps the first and second element of a set, $(\mathbf{12})\{a, b, c\} = \{b, a, c\}$ while keeping everything else the same. See Figure 2 for a pictorial illustration of this. We indicate the element $(\mathbf{i\ i{+}1})$, the permutation which swaps the ith and $i + 1$th element, as s_i.

Figure 2: s_1 acting on $\{a, b, c\}$

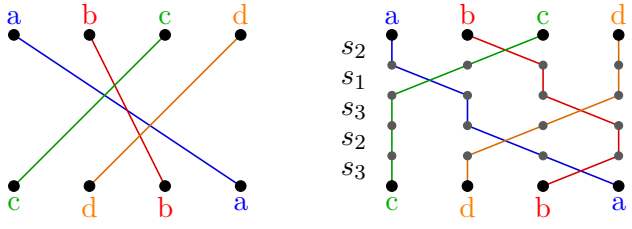


Interestingly, $\{s_i \mid 1 \leq i < n\}$ generates the group S_n. This means that given any permutation of a set, we can recreate it by repeatedly swapping two adjacent elements. As an example,

$$(\mathbf{1423})\{a, b, c, d\} = \{c, d, b, a\}$$
$$\textit{and}$$
$$s_3s_2s_3s_1s_2\{a, b, c, d\} = s_3s_2s_3s_1\{a, c, b, d\} = s_3s_2s_3\{c, a, b, d\} =$$
$$s_3s_2\{c, a, d, b\} = s_3\{c, d, a, b\} = \{c, d, b, a\}$$

We are able to pictorially represent this process using Figure 3. On the right, two adjacent elements are swapped one by one. While on the left, each element traces a direct path to its end location. One can trace the paths of all of the elements in each representation and find they end in the same location. This means that they are actually the same permutation!

Figure 3: (**1423**) and $s_3s_2s_3s_1s_2$ acting on $\{a, b, c, d\}$



Along with generating the group, all s_i follow a specific set of relations. We will now outline those relations. The first relation they follow is $s_i^2 =$

e. This says that applying s_i twice in a row is the same as doing nothing (Figure 4). They also follow the relation $s_i s_j = s_j s_i$ when $|i - j| > 1$. One can swap two pairs of elements in any order as long as the pairs are far enough apart (Figure 5). This is a secret braid relation, it is not typically called a braid relation but it is still a braid relation. Lastly, we have that $s_i s_{i+1} s_i = s_{i+1} s_i s_{i+1}$. This is our coveted braid relation from the introduction! This relation gives equivalent ways to swap three adjacent elements. It is the toughest to visualize but Figure 6 gives another visual representation.

Figure 4: s_1^2 and e acting on $\{a, b, c\}$

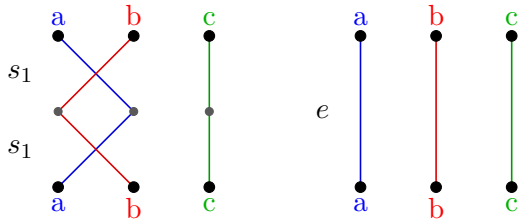
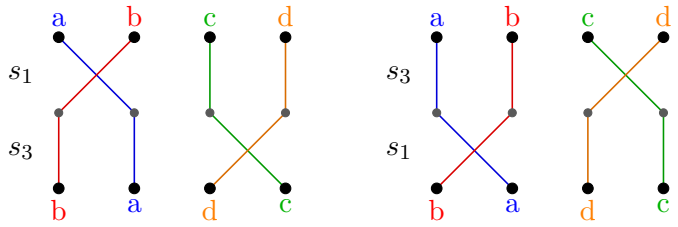


Figure 5: $s_1 s_3$ and $s_3 s_1$ acting on $\{a, b, c, d\}$

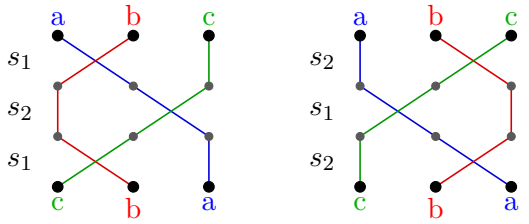


Given enough generators and relations, we can write a presentation for a group. A presentation is a neat way to completely describe a group. It encompasses all the elements of a group and their relationships in a small amount of easily legible ink. It follows the form:

$$\langle Generators | Relations \rangle$$

.

Figure 6: $s_1s_2s_1$ and $s_2s_1s_2$ acting on $\{a, b, c\}$



The generators, s_i, and the relations we described above are actually sufficient to write a presentation of S_n. This presentation is:

$$S_n = \left\langle s_1, s_2, \ldots, s_{n-1} | s_i^2 = e,\ s_is_j = s_js_i,\ s_is_{i+1}s_i = s_{i+1}s_is_{i+1} \right\rangle.$$

When one first sees this one might think presentations are a basic way of writing groups and thus they are not very exciting. This is not the case. In fact, their simplicity is what makes them so exciting.

The most basic way one can describe a group is by creating a table which shows the product of all elements of a group. This is called a Cayley Table. A Cayley table is very similar to a multiplication table. A Cayley Table of S_3 is given in Figure 7.

This might seem like a fine way to write stuff until one attempts to describe a group of meaningful size. By its tabular nature, there are ($Size\ of\ Group \times Size\ of\ Group$) entries in the table. The symmetric group, S_n, has size $n!$. Therefore there are $(n!)^2$ entries in its Cayley Table. This quickly becomes a detrimental amount of entries. In fact, there are $(6!)^2 = 518{,}400$ entries in the Cayley Table for S_6. Writing our presentation for S_6 is much less work and much more understandable than writing a practically innumerable amount of table entries. It is simply:

$$S_6 = \left\langle s_1, s_2, \ldots, s_5 | s_i^2 = e,\ s_is_j = s_js_i,\ s_is_{i+1}s_i = s_{i+1}s_is_{i+1} \right\rangle.$$

.

3 The Braid Group

We will now introduce the braid group from a symmetric group perspective and then quickly give a much more visual explanation. What happens when

Figure 7: A Cayley Table for S3, each element of the table is the product of its row and column group element

	e	s_1	s_2	s_1s_2	s_2s_1	$s_1s_2s_1$
e	e	s_1	s_2	s_1s_2	s_2s_1	$s_1s_2s_1$
s_1	s_1	e	s_2s_1	$s_1s_2s_1$	s_2	s_1s_2
s_2	s_2	s_1s_2	e	s_1	$s_1s_2s_1$	s_2s_1
s_1s_2	s_1s_2	s_2	$s_1s_2s_1$	s_2s_1	e	s_1
s_2s_1	s_2s_1	$s_1s_2s_1$	s_1	e	s_1s_2	s_2
$s_1s_2s_1$	$s_1s_2s_1$	s_2s_1	s_1s_2	s_2	s_1	e

we remove all "non-braid" relations from our presentation for the symmetric group? Will we break the space-time continuum? Solve the Riemann Hypothesis? No, but the answer is almost as astonishing. We get a presentation for the braid group,

$$S_n = \langle s_1, s_2, \ldots, s_{n-1} | s_i^2 = e,\ s_i s_j = s_j s_i,\ s_i s_{i+1} s_i = s_{i+1} s_i s_{i+1} \rangle$$
$$\cancel{S_n}\ B_n = \langle s_1, s_2, \ldots, s_{n-1} | \cancel{s_i^2 = e,}\ s_i s_j = s_j s_i,\ s_i s_{i+1} s_i = s_{i+1} s_i s_{i+1} \rangle$$
$$B_n = \langle \sigma_1, \sigma_2, \ldots, \sigma_{n-1} | \sigma_i \sigma_j = \sigma_j \sigma_i,\ \sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1} \rangle .$$

One can see from this presentation, that the braid group is actually of infinite size since $\sigma_1, \sigma_1\sigma_1, \sigma_1\sigma_1\sigma_1, \ldots$ are all unique elements in the group.

While this presentation is interesting to look at, it does not give us much intuition to what this group is actually like. For that, we need pictures and concrete examples. How can one picture an infinite group? Well the answer is actually quite intuitive and follows directly from our pictorial view of the symmetric group.

While S_n can be thought of as all sets of lines which send a collection of n objects to a rearrangement of those objects, we can picture B_n by transforming those lines into strings in three dimensions. Given a collection of objects and a duplicate collection of those objects, we can run strings from the first set of objects to the duplicate set of objects. A string runs from an object in the first set to an object in the second set. It can go in front of or behind other strings, but it can't go straight through another string. We also force the string to continuously move towards the second collection of objects, it can't move backwards. We do this to prevent creating any knots which would complicate our picture. We call each unique arrangement of strings a braid.

We call two braids the same if one can bend, stretch, or shrink the strings so that the braids are identical. In other words, we can move the strings around but can't uncross strings and can't switch which objects the strings are attached to. This concept is best understood through examples and pictures. Figure 8 and Figure 9 provide clear pictures of example braids in our group.

Figure 8: These two braids are equivalent. They can be bent into identical braids without uncrossing strings

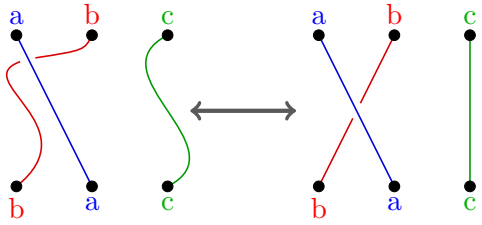
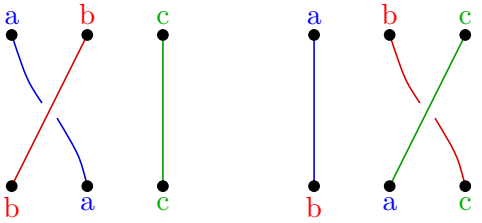


Figure 9: These two braids are different. There is no way to transform one into another without changing what objects the strings connect to.



What does σ_1 look like in this pictorial view? Well it is the braid which, while keeping all other strings straight, crosses the first string underneath the second string and thus also crosses the second string over the first string. Figure 10 illustrates this braid. Similarly, σ_i crosses the i string underneath the $i+1$ string while keeping all others straight. Since we no longer have that $\sigma_i^2 = e$, σ_i^{-1} must be its own unique element. In fact, it is the braid which crosses the i string underneath the $i+1$ string and keeps all others straight. Figure 11 illustrates how the composition of these elements is the same as doing nothing.

From our presentation we can infer that every possible braid is just a composition of these adjacent crosses. This is very non-obvious when one

Figure 10: σ_1 acting on $\{a, b, c\}$

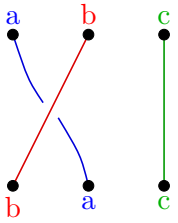
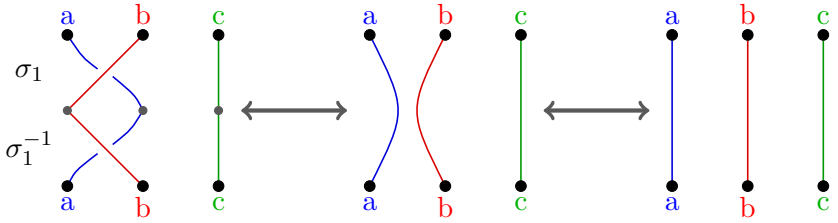


Figure 11: Through the power of bending we see that $\sigma_1\sigma_1^{-1}$ acts the same on $\{a, b, c\}$ as doing nothing



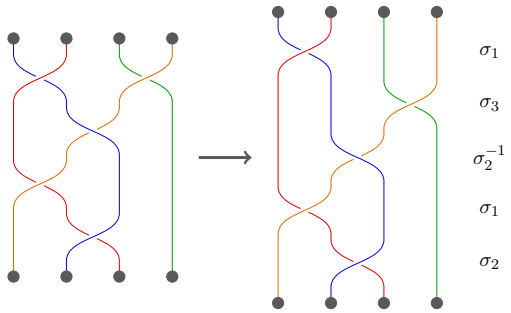
takes on the pictorial view of this group. A nice observation and clever use of stretching and shrinking allows us to give a reasonable argument as to why this is the case.

First notice that for two non adjacent string to cross each other, they must make a sequence of crosses past adjacent strings. There is no way two strings can cross each other without crossing the strings in-between them. Thus a braid is made up entirely of adjacent crossings. Since we can stretch the strings, stretch them up and down so that each crossing is in its own height segment with no other crossings at the same height. This braid now has the exact structure as a composition of adjacent crossings. See Figure 12 for a pictorial example of this process.

4 Applications

Knowing the basics structure of the braid group, we are now able to explore various applications of braid groups. We will look at two examples from very different fields of mathematics, cryptology and knot theory.

Figure 12: The process of separating a complicated braid to a slightly less complicated product of σ_is



4.1 Cryptology

Quantum computers are special. They are fundamentally different than today's computers. By taking advantage of principles like superposition, a quantum computer stores its information using qubits. A typical bit stores only 1's and 0's, true and false, while a qubit is able to store 1's, 0's, and the superpositional state of both 1 and 0, true, false and maybe.

This difference allows for quantum computers to be significantly better at certain tasks than non-quantum computers. It just so happens that solving the fundamental equations of many of today's cryptographic protocols is one of these tasks. These protocols encode information by using math problems that are very hard for normal computers to solve.

The most common math problem for a cryptology to be based on is that of factoring integers into their prime factors. An example of this would be:

$$28 = 2 \cdot 14 = 2 \cdot 2 \cdot 7.$$

When the numbers are much larger than 28, this is a very difficult thing to do. In fact it would take a traditional super computer more time than the universe has existed to brute force break a message using these cryptologies. Worryingly, factoring integers into prime factors falls under the tasks easy for a quantum computer to solve.

Because the world needs to encrypt stuff and effective quantum computers are an eventual reality, cryptologists have tried to create new protocols which are hard to break no matter the type of computer. One thing the protocols easily broken by quantum computers have in common is that they are based

on commutative groups. The integers is an example of a commutative group, the order of addition does not matter i.e. $2 + 3 = 6 = 3 + 2$. Thus new protocols are being developed based on non-commutative groups.

Fortunately, the braid group is non-commutative since $\sigma_i\sigma_{i+1} \neq \sigma_{i+1}\sigma_i$. Because of this fact, researchers have attempted to create a new protocol based on a hard to compute problem within the braid group[2][3]. An example of a hard problem within the braid group is the conjugacy search problem. The problem is as follows: *Given two braids, x and y in B_n, find a braid $s \in B_n$ so that $y = sxs^{-1}$.* It is known that no algorithm can compute s in under polynomial time regarding the size of n. This means that, like factoring integers, with a big enough n this problem is practically unsolvable. Because this is a hard problem relying on non-commutative groups, quantum computers will not solve it any easier.

4.2 Knot Theory

Knot Theory is quite simply, the study of mathematical knots. Mathematical knots differ slightly from what we might normally consider to be a knot. Mathematical knots are defined as a circle embedded in three dimensions. What this means is the knot has to have its ends connected. So Figure 13, which most people would call a knot, is not considered a mathematical knot since its ends do not connect. While if ones pocket was extra cluttered and they pulled Figure 14 out, it would be a mathematical knot since its ends are connected.

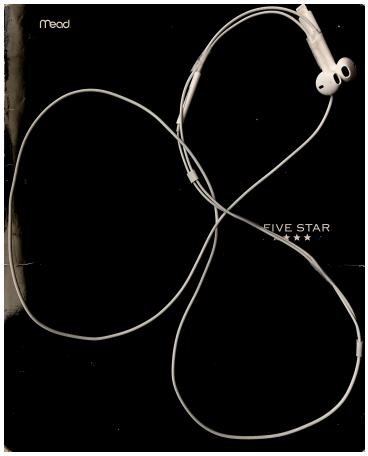
Mathematicians define knots this way because, like braids, they want to be able to bend and stretch knots without changing them. Since the ends of Figure 13 are disconnected, one could (with varying levels of frustration) untangle the knot back into a straight line. Meanwhile, it's impossible to untangle Figure 14, without breaking the tape.

How do I know that is impossible to untangle Figure 14? I know this because of knot theory! Figure 14 is actually the trefoil knot, the simplest knot that is not just a circle. One way to measure complexity of a knot is to count the minimum number of crossings it has. We need to count the *minimum* number of crossings because we are able to bend our knots. This would allow us to add any number of crossings we'd like, without changing the knot. See Figure 15 for an example of this.

Now that we understand complexity, a natural question to ask is, "How many knots have a specific number of crossings and what do they look like?".

Figure 13: A picture of my headphones after roughly six seconds in my pocket

Figure 14: A very messy pocket leads to a mathematical knot!



We have been able to fully classify knots of small numbers of crossings. See Figure 16 for a table of all knots with $0 - 7$ crossings. As the number of crossings increases, this becomes a complicated and difficult question.

One way we are able to better understand knots is through braids! By connecting the ends of the strands of a braid in a special way, we are able

Figure 15: While the trefoil has a minimum of three crossings, we can add as many extra crossings as we'd like

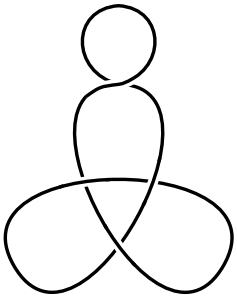
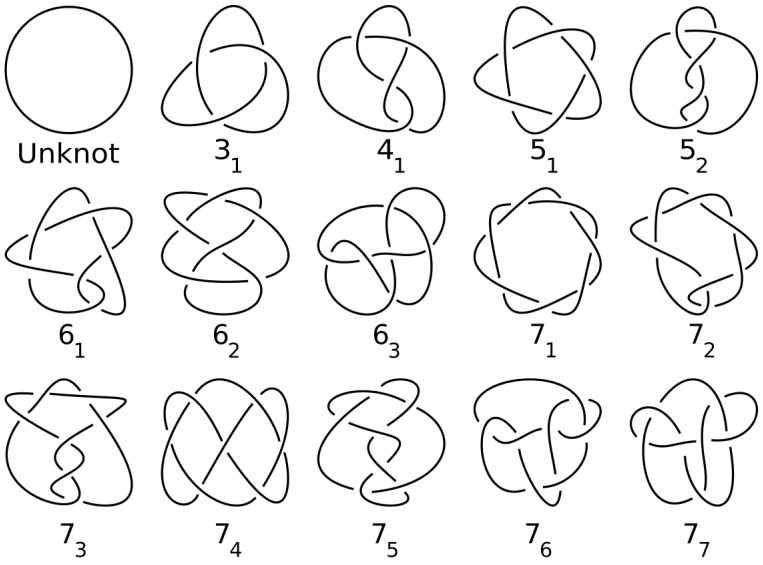
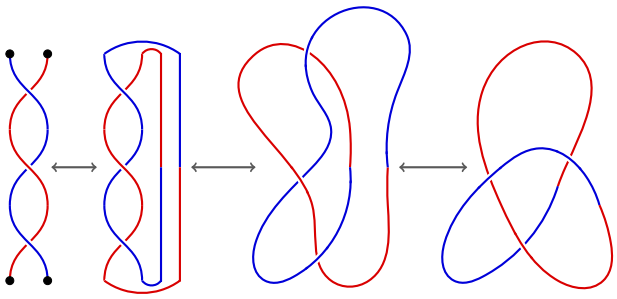


Figure 16: A knot table classifying all knots with 0-7 crossings



to form a knot. Figure 17 shows the process of closing up the braid $\sigma_1\sigma_1\sigma_1$ in order to form the trefoil knot. Its clear from this process that every braid can be formed into a knot.

Figure 17: The process of converting a braid into a knot



In fact, every knot can be formed as closure of a braid. This is an entirely non-trivial statement. It so non trivial that it actually has a name, Alexander's Theorem. It relies upon giving an orientation to a knot and then performing careful bends and twists to align segments of the knot so the orientation looks a specific way. By doing this, one actually forms a picture of a closed up braid. Since this process works for any knot, we've shown that every knot is a closed up braid.

This is a quite exciting fact, it gives us an avenue to use our knowledge of braids to better understand knots. The first knot invariant, which is a tool that can tell us if two knots are different, can actually be formed using properties of the braid group. This takes some more complicated mathematics, but the the key points are centered around our ability to write any braid as a product of σ_is and this transformation of braids into knots.

5 Acknowledgments

I am grateful to be able to thank my mentor, Vic Reiner, for all of his advice and teachings. Without him, braids and their braid relations would have just become a passing peculiarity for me. I'd also like to thank the University of Minnesota Twin-Cities McNair Team. I am incredibly grateful for the opportunity and funding to be able to spend a summer learning about the math I enjoy.

References

[1] A. Bjorner and F. Brenti. *Combinatorics of Coxeter Groups*. Graduate Texts in Mathematics. Springer Berlin Heidelberg, 2006. ISBN: 9783540275961. URL: https://books.google.com/books?id=1TBPz5sd8m0C.

[2] Xiaoming Chen et al. *A New Cryptosystem Based on Positive Braids*. 2019. arXiv: 1910.04346 [cs.CR].

[3] Ki Hyoung Ko et al. "New Public-Key Cryptosystem Using Braid Groups". In: *Advances in Cryptology — CRYPTO 2000*. Ed. by Mihir Bellare. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 166–183. ISBN: 978-3-540-44598-2.