Grade 9/10 Math Circles

An Introduction to Group Theory Part 3

Review from last time

Today we are going to learn about *braid groups*! But before we jump in, let's do a quick recap of what we have covered so far. Recall that a **group** is a set G, together with a binary operation $\bullet$, such that the following group axioms hold:

1. (Associativity) For every $a, b, c \in G$, $(a \bullet b) \bullet c = a \bullet (b \bullet c)$.
2. (Identity element) There exists $\text{id}_G \in G$ such that for all $a \in G$, $\text{id}_G \bullet a = a = a \bullet \text{id}_G$.
3. (Inverse element) For every $a \in G$, there exists $a^{-1} \in G$ such that $a \bullet a^{-1} = \text{id}_G = a^{-1} \bullet a$.

So far we have seen two main examples of groups. The first being symmetry groups, and the second being symmetric groups. Today we are going to look at another class of groups called *braid groups*!

Braid Groups

Like most groups, the study of *braid groups* has applications all over mathematics in areas such as topology, knot theory, physics, fluid mechanics, and algebraic geometry. Besides it's far reaching applications, braid groups are also visually nice and fun to play with because the elements in the underlying set of a braid group actually look like braids! Recall again that a group is a set together with a binary operation. Together, we will learn about the underlying set of a braid group. And throughout various exercises, you will figure out what the binary operation is!

The underlying set of a braid group The underlying set of a braid group is a certain collection of objects called *braids*. As hinted at earlier, these objects do in fact look like the braids you see in your everyday life. Here are some examples of braids out in the wild:

Figure 1: Examples of braids in everyday life.[1]

The braids that we are going to learn about today are just like the ones in Figure 1. Since we are doing mathematics though, we need to precisely define what we mean by a braid. For us, a **braid** is an object that is constructed via the step by step guide below.

[1]Belly Full. Challah Bread [Online]. Available from: https://bellyfull.net/challah-bread [Accessed April 4, 2023].

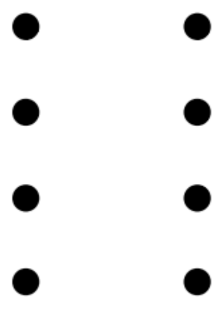
[1]Backyard Summer Camp. Easy Braided Friendship Bracelet Kids Craft [Online]. Available from: https://backyardsummercamp.com/braided-friendship-bracelet/[Accessed April 4, 2023].

[1]CBC. Orange-Infused Blueberry Pie with Braided Top [Online]. Available from: https://www.cbc.ca/life/video/orange-infused-blueberry-pie-with-braided-top-1.5134114 [Accessed April 4, 2023].

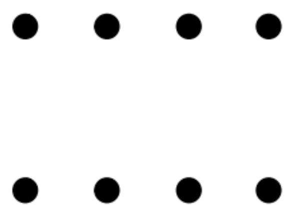
[1]Orvis. Classic Latigo Braid Belt [Online]. Available from: https://www.orvis.com/classic-latigo-braid-belt/2MFS.html [Accessed April 4, 2023].

Step by step guide for how to construct a braid:

1. Take any non-zero natural number n. In other words, pick $n \in \{1, 2, 3, \dots\}$.
2. Consider two collections of n dots. Arrange each collection of dots in a vertical line and put the collections side by side. Here is an example of how to arrange the dots when $n = 4$:



Note: If you prefer, you can instead arrange each collection of dots in a horizontal line, and then put one collection directly below the other. For $n = 4$*, this would look like:*



3. We use n strings to connect dots from one collection of dots to the other.
4. There are some rules we have to follow when connecting dots with strings:
   - Every dot from one collection must be connected to **exactly** one dot from the other collection.
   - Knots are **not** allowed.

Let's look at some examples of braids when $n = 4$.

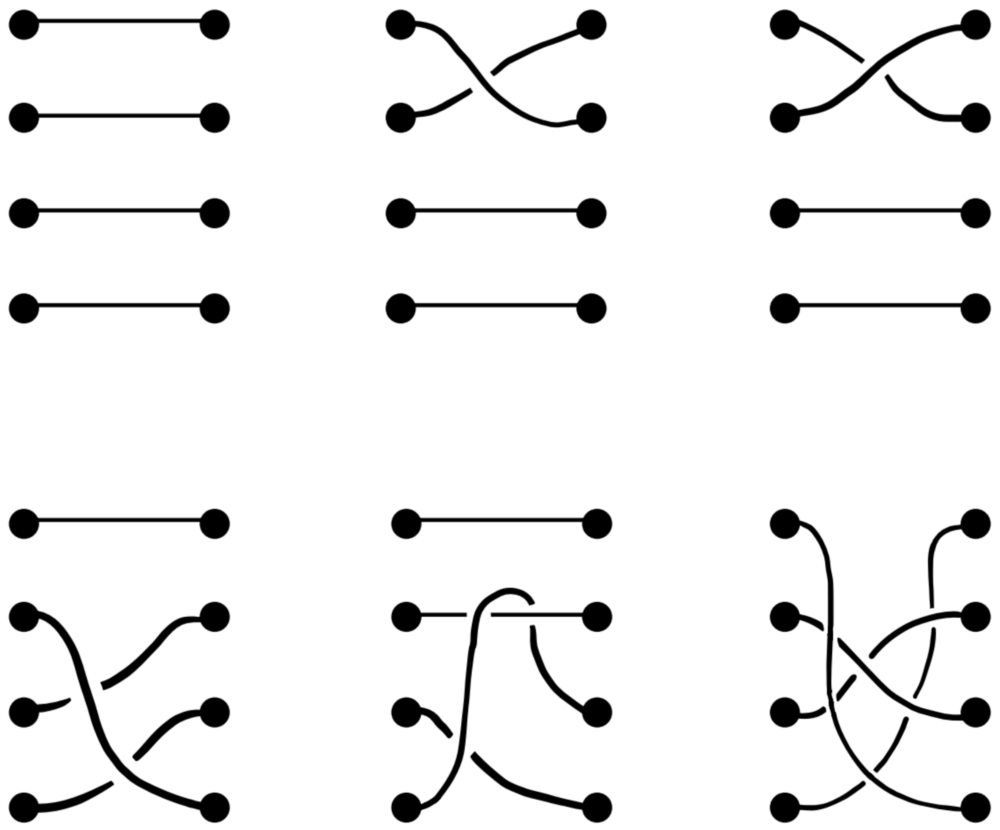
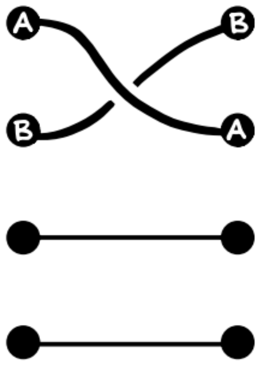
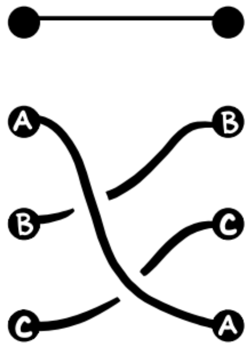


Figure 2: Examples of braids.

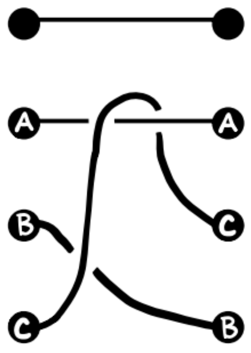
All of the diagrams in Figure 2 are examples of braids. In some of the diagrams, you may have noticed that there are "spaces" or "breaks" in some of the strings. Let's explain this by first looking at the second braid:



In this diagram, we see that there are spaces in the string connecting the "B" dots. These spaces are used to indicate that the string connecting the "A" dots lies on top of the string connecting the "B" dots at that spot. In other words, the string connecting the "A" dots crosses over, or goes over, the string connecting the "B" dots. Let's look at another braid:



The spaces here tell us where the string connecting the "A" dots crosses over the string connecting the "B" dots and the string connecting the "C" dots. For another example, consider the following braid:



The spaces here are a bit more involved than in the braids above. First off, the string connecting the "C" dots crosses over the string connecting the B" dots. We also see that the string connecting the "C" dots goes over the string connecting the "A" dots, and then goes underneath it. This is not a knot, and so is allowed. Let's now look at some non-examples:

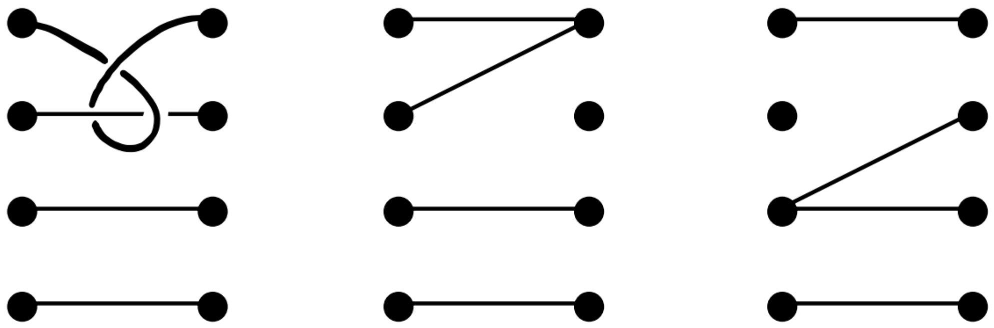
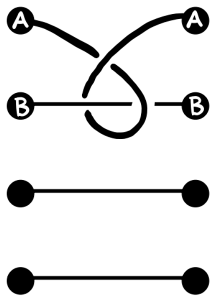


Figure 3: Non-examples of braids.

All of the diagrams in Figure 3 are **not** braids. In the last two diagrams, we can see that not every dot is connected to a string (and as a result, there are dots connected to more than one string), and so they are not braids. Now consider the first diagram in Figure 3:



Here, every dot is connected to exactly one string, however, there is a knot! The string connecting the "A" dots makes a knot around the string connecting the "B" dots. So, this is not a braid.

Definition 1

The objects formed using the above "step by step guide for how to construct a braid" are called **braids**.

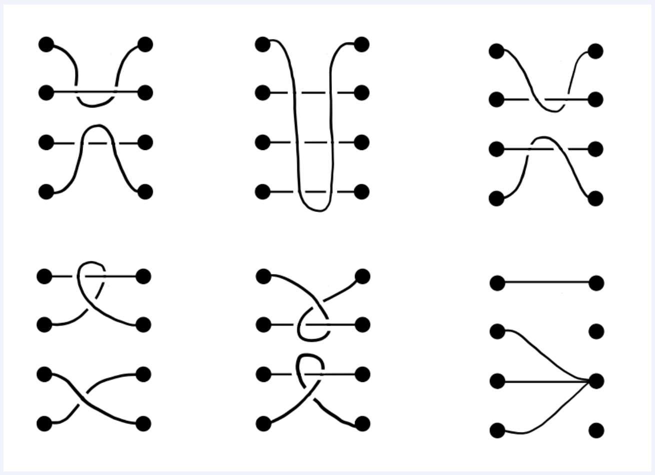
Definition 2

Let $n \in \{1, 2, 3, \ldots\}$. A braid formed using n strings is called an ***n*-braid**. We use $\boldsymbol{B_n}$ to denote the set of all n-braids.

For each n, the set B_n together with a certain binary operation forms a group. This group is called the **braid group on** n **strings**. We will come back to this soon!

Exercise 1

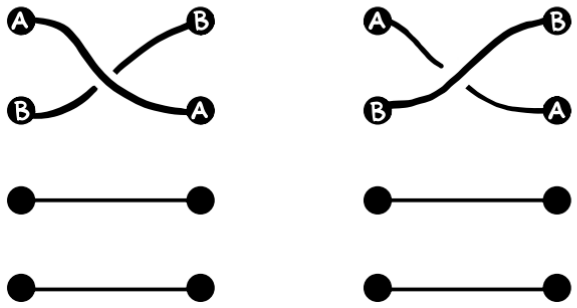
Identify which of the below diagrams are 4-braids.



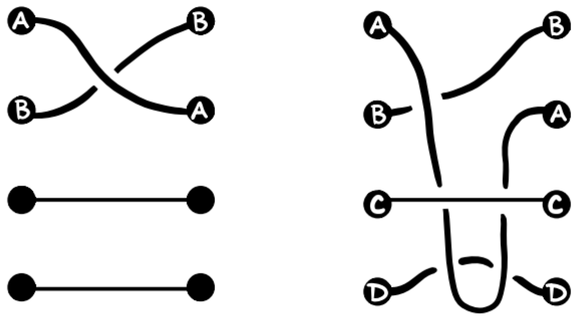
Exercise 2

Choose a small n, say $1 \leq n \leq 6$, and make n-braids. Feel free to use a few different n values.

Sometimes two or more n-braids appear to be different, but they are actually the same braid. Let's look at some examples. Consider the following two braids:



These two braids are different or, in other words, distinct. Note that the bottom two string connections in each braid are identical, so we only need to focus on the top two. In the first braid, the string connecting the "A" dots crosses over the string connecting the "B" dots. In the second braid, the opposite happens; the string connecting the “B” dots crosses over the string connecting the “A” dots. Given this, these two braids are different. Now, consider these two braids:

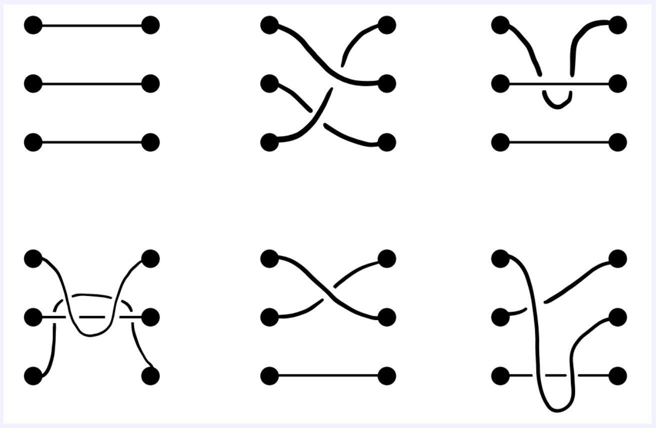


These two braids look quite different from each other, but they are actually the same braid! We can see this by “pulling” and “stretching” the strings. Let’s start by looking at the strings that connect the “A” dots and the “B” dots. Our goal is to make these strings look the same on each braid by pulling and stretching them. To do this, let’s make the second braid look like the first. Note that the string connecting the “A” dots in the second braid lies on top of the string connecting the “D” dots, and lies below the string connecting the “C” dots. Because there is no intertwining or crossings between these strings, we can pull the string connecting the “A” dots towards the top of the braid; it will slide over the string connecting the “D” dots and slide underneath the string connecting the “C” dots. After doing this, we can see that the strings connecting the “A” dots and “B” dots are the same on each braid. Also, after doing this, we can see that the string connecting the “C” dots is the same on both braids, and similarly, the string connecting the “D” dots is the same on each braid. Given this, the two braids are considered to be the same! Note that for the example on page 8, there is no way to pull and stretch the strings to make them the same on each braid; this is because pulling and stretching strings cannot undo or create crossings.

To conclude, if two n-braids can be made to look the same by “pulling or stretching the strings”, then we say that the two n-braids are the **same**. In other words, if you can change, or transform, one braid into another (by pulling or stretching the strings) without moving the starting and end points of the strings, then the two braids are the same. If two n-braids are not the same, then they are **different**. Note that the length of strings does not matter when in comes to determining if two braids are the same or different.

Exercise 3

Consider the six 3-braids below. How many different 3-braids are there?



Binary operation on braids So far we defined the underlying set for braids groups. Specifically, for each non-zero $n \in \mathbb{N}$, there is a group called the braid group on n strings, whose underlying set is B_n. We still need a binary operation on B_n to make it into a group! Now that you have some practice working with braids, you are going to try to figure out what the binary operation on B_n is.

Exercise 4 **∗∗ Spoilers on the next page∗∗**

Try to define a binary operation $\bullet$ on B_n so that $(B_n, \bullet)$ is a group. Here are some suggestions and comments to help you get started:

- Asking for a binary operation on B_n is the same as asking "how can we combine two n-braids to make another n-braid?".
- Feel free to use a small n, say $n = 3$ or $n = 4$, to figure out the binary operation.
- Don't worry too much about trying to formalize the binary operation. Playing around and trying to combine explicit n-braids in a natural way is a great way to get you on the right track!
- Keep in mind that we want this binary operation and B_n to form a group. Once you have a guess of what the binary operation on B_n is, try to see if the group axioms hold for B_n and your binary operation.

Consider the set B_n consisting of all n-braids. We can combine two n-braids in B_n by a binary operation called *concatenation*. We use the symbol $*$ for concatenation.

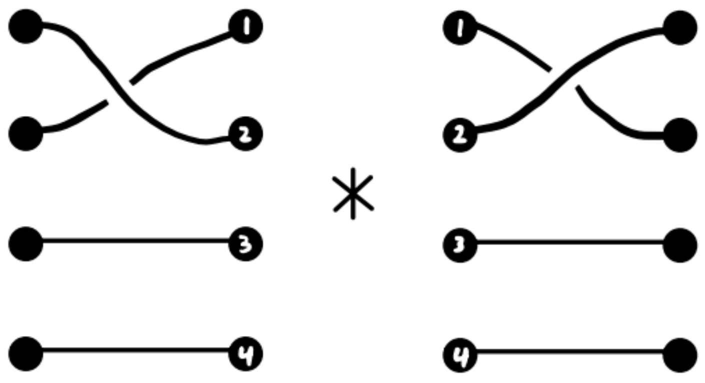
Definition 3

Let b_1 and b_2 be two n-braids. Then $b_1 * b_2$ is a new n-braid which is formed by connecting the right set of dots in b_1 to the left set of dots in b_2 and then erasing these dots so that the strings from the two braids form a connection. Specifically, if we label each set of n dots 1 through n from top to bottom, then we connect the kth dot from the right set of dots in b_1 to the kth dot in the left set of dots in b_2. This is a binary operation on B_n called **concatenation**.

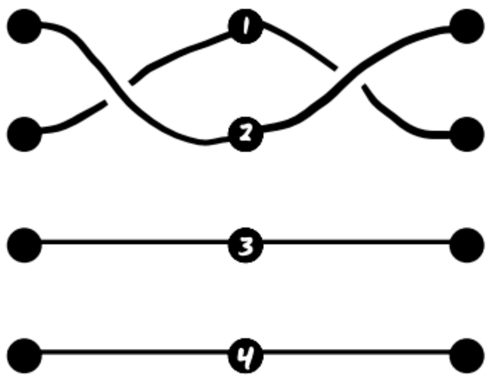
Note

In general, given two n-braids b_1 and b_2, $b_1 * b_2$ is not necessarily equal to $b_2 * b_1$.

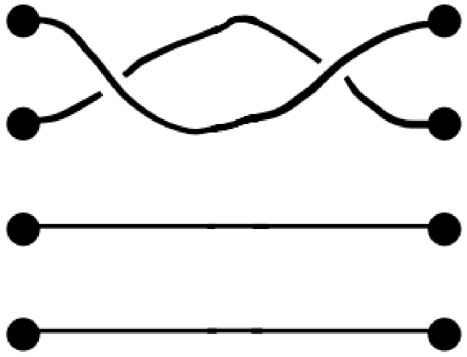
For an example, let's compute the following concatenation:



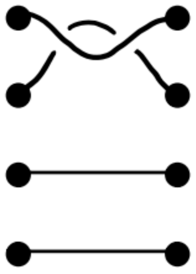
As suggested in Definition 3, you can see that we labelled two sets of dots 1 through 4. This is completely optional, and is just used as a way to help understand how concatenation works. To compute this concatenation, we connect dot 1 in the left braid with dot 1 in the second braid, dot 2 in the first braid with dot 2 in the second braid, and so on. The result is as follows:



To make this into a 4-braid, we simply erase the middle set of dots. The result is as follows:



We can also shorten the strings to make the diagram tidier. The result is as follows:



In short, to compute $b_1 * b_2$, you can imagine sliding the two braids together so that the right set of dots in b_1 lines up with the left set of dots in b_2. Next, we will convince ourselves that B_n under concatenation $*$ forms a group. In other words, we will convince ourselves that the group axioms hold for B_n with $*$.

Exercise 5

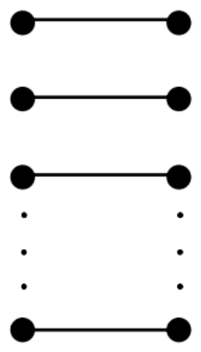
Convince yourself that $(B_3, *)$ is a group.

$**$ See solutions to Exercise 5 before reading on $**$

The explanation for Exercise 5 can be generalized to show that $(B_n, *)$ is a group for all non-zero $n \in \mathbb{N}$. Let's roughly see why $(B_n, *)$ is a group for any n. To do this, we need to convince ourselves that the group axioms hold for B_n with concatenation.

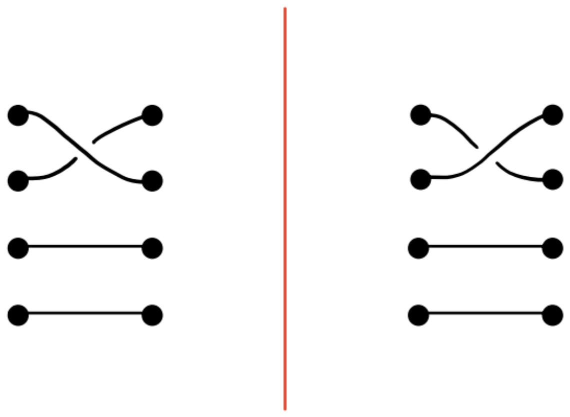
Axiom 1: For associativity to hold, we need $(b_1 * b_2) * b_3$ to be the same n-braid as $b_1 * (b_2 * b_3)$, for all $b_1, b_2, b_3 \in B_3$. This axiom is a bit hard to argue rigorously without introducing a lot of extra notation. However, for us, it is plenty good enough to understand why it works by looking at concrete examples. See the solutions to Exercise 5 for a concrete example worked out.

Axiom 2: The identity element in B_n is the following braid:

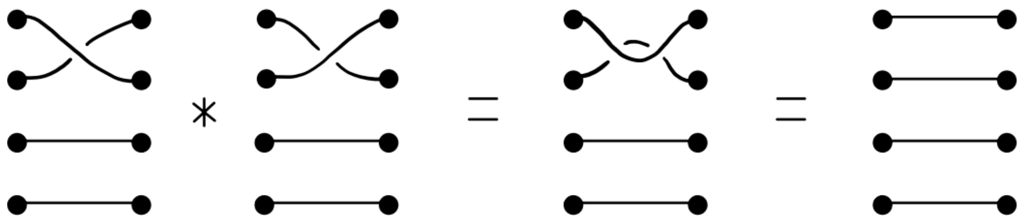


This braid consists of n parallel horizontal strings. If you concatenate this braid with any other n-braid b, the strings of b just get longer. Since the length of strings does not matter, the result is just b. So, this braid is the identity element and we call it id_{B_n}.

Axiom 3: Lastly, we want to show that every n-braid has an inverse. The inverse of an n-braid is obtained by flipping it over horizontally. For a concrete example, consider the following photo:



Call the braid on the left b. Flipping b over horizontally is the same as reflecting b in the red vertical line. To obtain the inverse of b, all we have to do is reflect b in this line. The braid on the right is what we get after reflecting b in this line. We will soon see that the braid on the right is the inverse of b, so let's call it b^{-1}. Indeed, we compute that $b * b^{-1}$ is



By stretching and pulling on the strands of $b * b^{-1}$, we see that it is equal to id_{B_n}. In the same way, one can show that $b^{-1} * b = \text{id}_{B_n}$. So b^{-1} is the inverse of b. And in general, the inverse of any n-braid can be obtained by using the method illustrated in this example. If you think of an n-braid as a certain "tangling" of id_{B_n}, then it's inverse is the n-braid that undoes, or is the reverse, of this tangling.

Definition 4

We call the group $(B_n, *)$ the **braid group on n strands**.