Combinatorics Notes

Trevor Gunn

2024-03-08

Table of contents

Preface

Combinatorics notes extending the book Applied Combinatorics by Keller and Trotter.

1 Inclusion/Exclusion

1.1 For two sets

Example 1.1. Suppose there are 100 CS students an a school. Of these, 50 know Java and 60 know Python (everyone knows at least one of these two languages). If we add $50 + 60$ we get the 100 total students plus 10 students who know both.

$$50 + 60 = 100 + 10.$$

Given two sets A and B, we can compute $|A \cup B|$ by first adding everything in A and adding everything in B. This counts all of $A \cup B$ except the items in $A \cap B$ were counted twice. Thus

$$|A| + |B| = |A \cup B| + |A \cap B|.$$

1.2 For three sets

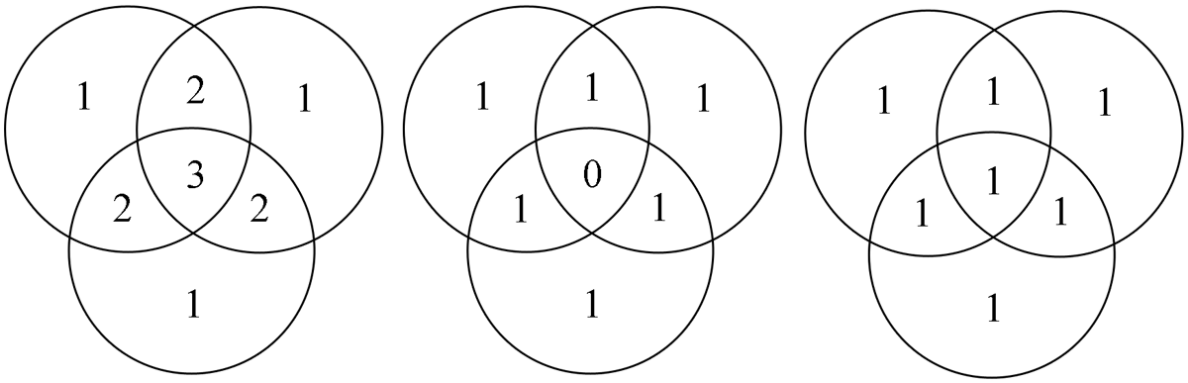


Figure 1.1: Inclusion/Exclusion for 3 sets

We can do a similar process for three sets. Start by adding up $|A| + |B| + |C|$ then subtract all the intersections of pairs then add back in $|A \cap B \cap C|$. For items in each region of the Venn diagram, think about how many times an item is added/subtracted overall.

E.g. items in A and B but not C will be added in twice in the first step ($|A| + |B|$) and subtracted once in the second step ($|A \cap B|$) so they are counted once.

Items in $A \cap B \cap C$ are added in $3 - 3 + 1 = 1$ times.

1.3 Binomial coefficients and I/E

Suppose we are looking at a union of n sets: $A_1 \cup \cdots \cup A_n$. Consider an element x belonging exactly to $A_1, \ldots, A_m$ and no other sets. If we do the same procedure of adding $|A_1| + \cdots + |A_n|$ then we are counting x a total of m times. Then, when we subtract all the pairwise intersections, $|A_i \cap A_j|$, we are subtracting from the count of x a total of $\binom{m}{2}$ because there are $\binom{m}{2}$ ways to choose two indices $\{i, j\} \subset \{1, \ldots, m\}$.

Thus, if we alternate: adding in single sets, subtracting pairs of intersections, adding tripples, subtracting quadruples, etc. we are counting x a total of

$$\binom{m}{1} - \binom{m}{2} + \binom{m}{3} - \binom{m}{4} + \cdots \tag{1.1}$$

times.

Let's have a closer look at this. We know the Binomial Theorem says that

$$(1 + x)^m = \binom{m}{0}x^0 + \binom{m}{1}x^1 + \binom{m}{2}x^2 + \cdots + \binom{m}{m}x^m.$$

Substituting $x = -1$, we get

$$0 = \binom{m}{0} - \binom{m}{1} + \binom{m}{2} - \cdots + \binom{m}{m}(-1)^m.$$

And since $\binom{m}{0} = 1$, if we move all the other terms to the other side of the equation, we see that Equation 1.1 evaluates to 1.

> **Note 1**
>
> For this problem we are counting the size of $A_1 \cup \cdots \cup A_n$ so each element is in at least one set ($m \geq 1$). This is important so that we can say that $0^m = 0$. In the context of the binomial theorem, $(1 - 1)^0 = \binom{0}{0} = 1$. This will be important later when we want to count elements not belonging to any set.

The general I/E formula is then

$$\begin{aligned}|A_1 \cup \cdots \cup A_n| &= \sum_i |A_i| - \sum_{i<j} |A_i \cap A_j| + \sum_{i<j<k} |A_i \cap A_j \cap A_k| - \dots \\ &= \sum_{t=1}^{n} (-1)^{t+1} \sum_{i_1<\cdots<i_t} |A_{i_1} \cap \cdots \cap A_{i_t}|.\end{aligned}$$

Note: $(-1)^{t+1}$ takes the values $1, -1, 1, -1, \dots$ starting at $t = 1$.

1.4 Avoiding properties

Inclusion/Exclusion appears most frequently in combinatorics not as a means to count $|A_1 \cup \cdots \cup A_n|$ directly but rather as a means to count everything *not* in $A_1 \cup \cdots \cup A_n$.

For example, let \$[m] = {1,...,m} and suppose we want to count the number of functions $f : [m] \to [n]$ which *don't* miss anything in the codomain. I.e. if A_i is the set of functions where $f(x)$ never equals i, then we want to count every function *not* in any set A_i.

Let X be the total set of functions. Then to count the functions avoiding $A_1 \cup \cdots \cup A_n$ we do

$$\begin{aligned}|X \setminus (A_1 \cup \dots A_n)| &= |X| - |A_1 \cup \dots A_n| \\ &= |X| - \sum_i |A_i| + \sum_{i<j} |A_i \cap A_j| - \cdots\end{aligned}$$

> **Caution**
>
> The formula $|A \setminus B| = |A| - |B|$ works only when B is contained entirely inside A.

1.5 Simplifying notation.

Let $\mathscr{P} = \{P_1, \dots, P_n\}$ be a collection of sets representing negative properties—conditions we want to avoid. For a subset $S \subseteq \{1, \dots, n\}$, let $N_{\geq}(S)$ be the number of items satisfying at least those properties in S. I.e. $\bigcap_{i \in S} P_i$. We will say "x satisfies S" for short. Also, if $S = \{a, b, c\}$, let us write for example, $N_{\geq}(a, b, c)$ instead of $N_{\geq}(\{a, b, c\})$ for simplicity.

Let $N_{=}(S)$ be the number of x satisfying exactly those properties in S and no properties not in S. When $\mathscr{P}$ represents properties we wish to avoid, then $N_{=}(\varnothing)$ is the number of elements satisfying none of the properties.

Theorem 1.1 (Inclusion/Exclusion)**.**

$$\begin{aligned} N_{=}(\varnothing) &= N_{\geq}(\varnothing) - \sum_i N_{\geq}(i) + \sum_{i<j} N_{\geq}(i,j) - \cdots \\ &= \sum_S (-1)^{|S|} N_{\geq}(S). \end{aligned}$$

Proof. As in Section 1.3, we will break up the sum focusing on each element. So first, we will write

$$\sum_S (-1)^{|S|} N_{\geq}(S) = \sum_S (-1)^{|S|} \sum_{\substack{x \\ x \text{ satisfies } S}} 1$$

Here we replace the number $N_{\geq}(S)$ by a count of 1 for each element x which satisfies S. This introduces a second sum into the picture and that will allow us to swap the order of summations. Currently, the second sum is over all x with respect to the relation "x satisfies S." When we put the sum over x outside, we still have the relation "x satisfies S" but rather than summing over all x with this property, we sum over all S:

$$\sum_S (-1)^{|S|} \sum_{\substack{x \\ x \text{ satisfies } S}} 1 = \sum_x \sum_{\substack{S \\ x \text{ satisfies } S}} (-1)^{|S|}.$$

We also move the $(-1)^|S|$ inside the second sum because that quantity depends on $|S|$.

Next, just as we did in Section 1.3, let us say that x satisfies exactly $P_{i_1}, \ldots, P_{i_m}$ (m will depend on x) and let $S_x = \{i_1, \ldots, i_m\}$. This set is the largest set that x satisfies. Every other set that x satisfies will be a subset of S_x.

We now break up our sum based on the size of those subsets $S \subseteq S_x$, using the binomial coefficients to count the number of such S:

$$\sum_x \sum_{\substack{S \\ x \text{ satisfies } S}} (-1)^{|S|} = \sum_x \sum_{k=0}^{m} \underbrace{\binom{m}{k} (-1)^k}_{\substack{|S|=k \text{ and } x \text{ satisfies } S \\ \iff |S|=k \text{ and } S \subseteq S_x}}.$$

This, by the Binomial Theorem, is the same as

$$\sum_x (1-1)^m.$$

Remember here that m depends on x.

As discussed in Note 1, $0^m = 0$ if $m \geq 1$ but if $m = 0$ then $0^0 = 1$. Saying $m = 0$ means that x satisfies exactly 0 properties—which is what we are looking for. So the final simplification looks like

$$\sum_x 0^m = \sum_{\substack{x \\ m=0}} 1 = N_=(\varnothing).$$

□

1.6 Application 1: Surjections

As in Section 1.4, let X be the set of all functions from $[m]$ to $[n]$ and let P_i be the set of functions where i is never an output: $f(x) \neq i$ for any input x.

Lemma 1.1.

a) The number of functions from an m element set to an n element set is n^m.
b) The number of functions from an m element set to an n element set that avoid k outputs is $(n-k)^m$.

Proof.

a) There are n choices for $f(1)$ and n choices for $f(2)$, etc. So there are n^m choices in total for $f(1), \ldots, f(m)$.
b) If we are avoiding k outputs then there are only $n - k$ choices for each of $f(1), \ldots, f(m)$ so $(n-k)^m$.

□

With this in mind, we have $N_{\geq}(S) = (n-k)^m$ if $|S| = k$ (we avoid at least the specified k outputs). So by inclusion exclusion, the number of functions which avoid *no* outputs (i.e. surjections) is

$$\begin{aligned} &N_{\geq}(\varnothing) - \sum_i N_{\geq}(i) + \sum_{i<j} N_{\geq}(i,j) - \cdots \\ &\qquad = n^m - \sum_i (n-1)^m + \sum_{i<j} (n-2)^m - \cdots . \end{aligned}$$

And since there are $\binom{n}{k}$ ways to choose k outputs to avoid, we can also write this as

$$n^m - \binom{n}{1}(n-1)^m + \binom{n}{2}(n-2)^m - \cdots = \sum_k \binom{n}{k}(-1)^k (n-k)^m.$$

The textbook calls this number $S(m, n)$.

> **Note**
>
> In the above application, $N_{\geq}(S) = (n-k)^m$ only depended on the size k of S. This is true in many but not all examples.

1.7 Application 2: Derangements

Let X be the set of all permutations of $[n]$. We wish to count the permutations with no fixed points. So let P_i be the property that i is a fixed point. Then $N_{=}(\varnothing)$ is the number of permutations with zero fixed points. This number is called d_n in the textbook.

Lemma 1.2.

a) The number of permutations of $[n]$ is $n!$
b) The number of permutations with at least k specified fixed points is $(n-k)!$

Proof.

a) Discussed in Chapter 2 of the book.
b) To say that there are k specified fixed points means were are permuting the other $n-k$ items. Similar to (a), there are $(n-k)!$ ways to permute $n-k$ items.

□

Applying Inclusion/Exclusion, we thus have

$$d_n = n! - \binom{n}{1}(n-1)! + \binom{n}{2}(n-2)! - \cdots = \sum_k \binom{n}{k}(-1)^k(n-k)!.$$

Again, there are $\binom{n}{k}$ ways to choose a set S of k fixed points and each of these has the same number $N_{\geq}(S) = (n-k)!$.

2 Permutations

A permutation of a finite set X can be thought of in a few ways. The set of all permutations of $\{1, \ldots, n\}$ will be denoted S_n. This is also called *the symmetric group*.

2.1 As a shuffle of the symbols

E.g. 53214 is a shuffle of $X = \{1, 2, 3, 4, 5\}$.

This gives us one way of counting the number of permutations. For the first position in the shuffle, we may write any of the n numbers. For the second position, we have $n - 1$ numbers to choose from—we cannot repeat the first. Likewise the third has $n - 2$ to choose from, not repeating the first or second. In this way, the number of permutations is

$$n \cdot (n-1) \cdot (n-2) \cdots 2 \cdot 1, \text{ denoted } n!.$$

2.2 As a function from X to itself

E.g. The shuffle 53214 may be thought of as a function $\pi : X \to X$ where $\pi(i)$ equals the i-th number in the shuffle:

$$\pi(1) = 5, \pi(2) = 3, \pi(3) = 2, \pi(4) = 1, \pi(5) = 4.$$

These functions are often represented by a table:

i	1	2	3	4	5
$\pi(i)$	5	3	2	1	4

Something new added by the table/function representation is that we can talk about the inverse function which undoes the shuffle:

$j = \pi(i)$	5	3	2	1	4
$i = \pi^{-1}(j)$	1	2	3	4	5

or reordering the columns:

j	1	2	3	4	5
$\pi^{-1}(j)$	4	3	2	5	1

2.2.1 Composition of functions

Given two permutations, $\pi_1, \pi_2 : X \to X$, we can compose them to get a new function $(\pi_1 \circ \pi_2)$. One way to compute this is via the following procedure:

1. Write the two functions as tables.
2. Reorder the columns of the outermost function in the composition to align with the output of the innermost function.
3. Stack the tables on top of eachother.

E.g. Take π_2 to be the function represented by the shuffle 53214 and take π_1 to be represented by the shuffle 13254. So as a table

i	1	2	3	4	5
$\pi_1(i)$	1	3	2	5	4

Now we shuffle the columns so the top is 53214:

i	5	3	2	1	4
$\pi_1(i)$	4	2	3	1	5

Then stack this with π_2:

i	1	2	3	4	5
$j = \pi_2(i)$	5	3	2	1	4
$\pi_1(j)$	4	2	3	1	5

The bottom row is $\pi_1 \circ \pi_2$.

Exercise 2.1. Compute this in the other order: $\pi_2 \circ \pi_1$. *Observe that the order matters.*

Note: in some sources (e.g. books, software), compositions are witten as a product in the reverse order. Here are computation in the software SageMath.

```
π1 = Permutation([1,3,2,5,4])
π2 = Permutation([5,3,2,1,4])
π2 * π1
```

```
[4, 2, 3, 1, 5]
```

```
π1 * π2
```

```
[5, 2, 3, 4, 1]
```

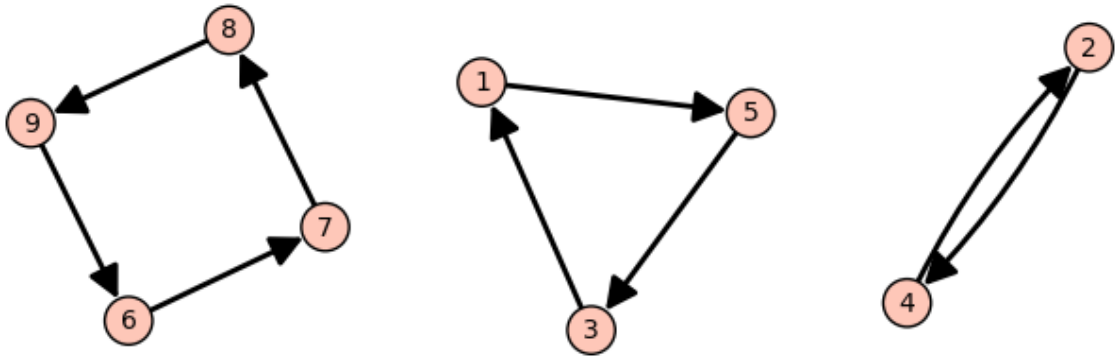
3 Visual representations of permutations

3.1 As a collection of cycles

Let's look at a longer permutation now: 541237896. As in Section 2.2, we can view this as a function where 1 goes to 5, 2 goes to 4 etc. Following the path of a single number we get a cycle: 1 goes to 5 goes to 3 goes to 1. Likewise 2 goes to 4 goes to 2. And so on.

The entire permutation can be broken up into cycles:

```
π = Permutation([5,4,1,2,3,7,8,9,6])
π.to_digraph().plot(vertex_size=1500)
```



3.1.1 Notation

The cycle $1 \to 5 \to 3 \to 1$ can be written as (153). In general, $(a_1, \ldots, a_n)$ represents the cycle $a_1 \to a_2 \to \cdots \to a_n \to a_1$. We often ommit the commas when every number is a single digit.

Here is how to compute the cycles in SageMath:

```
π.cycle_string()
```

```
'(1,5,3)(2,4)(6,7,8,9)'
```

The answer is given as a product (i.e. composition) of cycles.

Elements which do not go anywhere—also called *fixed-points*—are represented by cycles of length 1 E.g. $(123)(4)$ represents the two cycles $1 \to 2 \to 3 \to 1$ and $4 \to 4$. We often ommit length 1 cycles from the notation so $(123)(4) = (123)$ as a permutation of $1, 2, 3, 4$.

Exercise 3.1. Compute the cycle decomposition for 341859672. You can verify your answer in SageMath.

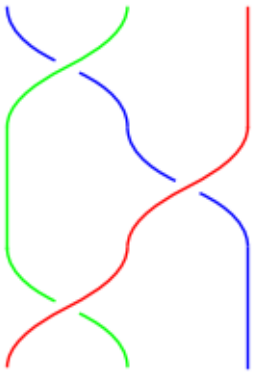
```
π = Permutation([3,4,1,8,5,9,6,7,2])
π.cycle_string()
```

Note: as mentioned earlier: SageMath ommits the cycle (5) representing the fixed point $\pi(5) = 5$.

3.2 Braids

Another visual representation used frequently in the mathematical study of knots is that of crossing lines. Let's look at an example:

```
B.<a,b> = BraidGroup(3)
plot(a * b * a)
```



This represents a permutation where the first end ends up in the third position ($1 \to 3$), the second end ends up in the second position ($2 \to 2$) and the third end ends up in the first position ($3 \to 1$). Overall, 1 and 3 are swapped, so this permutation is (13).

3.2.1 Transpositions

Cycles of length 2 are called *transpositions*. E.g. (14) is a transposition which switches 1 and 4. The picture above represents a composition of 3 transpositions (13) = (12)(23)(12).

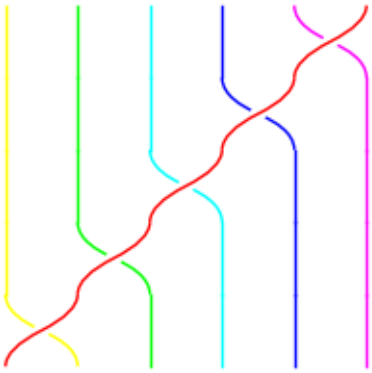
Theorem 3.1 (Product of transpositions)**.**

a) Every permutation can be written as a product of transpositions
b) Every permutation can be written as a product of transpositions of adjacent elements.

Another way to say this is that by swapping pairs of elements at a time, we can obtain any possible shuffle.

Proof. We know that a permutation may be written in terms of cycles. So if we can show that any cycle can be written as a successive sequence of swaps we are good. We will give a visual demonstration of this fact:

```
B.<t1,t2,t3,t4,t5> = BraidGroup(6)
plot(t1 * t2 * t3 * t4 * t5)
```



So the cycle (123456) = (12)(23)(34)(45)(56). Remember that compositions are read from right to left: e.g. $(f \circ g)(i) = f(g(i))$ means first do g then do f.

This kind of decomposition generalizes: you can replace (123456) with any cycle of any length. E.g. (1456) = (14)(45)(56) although one more reminder that products in SageMath are backwards from the function composition standpoint:

```
S = SymmetricGroup(6)
S((5,6)) * S((4,5)) * S((1,4))
```

```
(1,4,5,6)
```

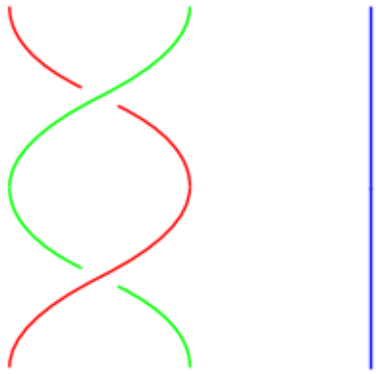
□

For an alternative proof, this decomposition into a sequence of adjacent swaps is exactly how the sorting algorithm BubbleSort works. We can sort any list using BubbleSort which does only adjacent swaps. So the shuffle is obtained by reversing those swaps to go from sorted to shuffled.

3.2.2 Braids versus permutations

You may have noticed in what we did, the word "braid" was used. Braids are similar to permutation except that we keep track of which strand goes above and which strand goes below. E.g. the transposition (12) applied twice looks like

```
B.<a,b> = BraidGroup(3)
plot(a * a)
```



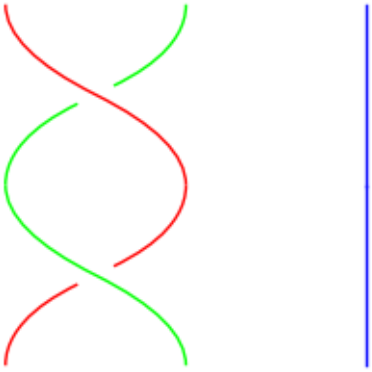
As a permutation this is the trivial shuffle 123. But as a braid it is still twisted.

While we are only focused on permutations rather than braids here, we still make use of braids because SageMath is able to create diagrams for us.

3.2.2.1 The Braid Group in SageMath

The generators of the braid group are adjacent swaps. So the line `B.<a, b> = BraidGroup(3)` sets `a` to a swap of $1, 2$ and sets `b` to a swap of $2, 3$. We can obtain the reverse swap with `a^-1` or `b^-1`

```
plot(a^-1 * a^-1)
```



When converting from braids to permutations, we ignore whether a strand goes over or under and just focus on the swapping.

4 Generators

In Theorem 3.1 we saw that every cycle and hence every permutation can be written as a product of adjacent swaps: $(12), (23), (34), \ldots, (n-1, n)$. Here we will identify some relations that hold among these generating elements.

Let us call $\tau_i = (i, i+1)$ the swap of i and $i+1$.

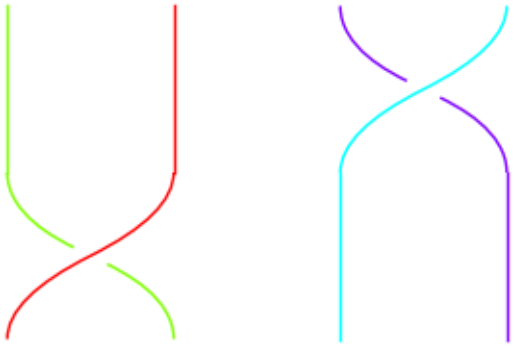
4.1 Squaring relation

We have $\tau_i^2 = 1$ where 1 represents the identity permutation (no shuffling). This says that if we swap i and $i+1$ and then swap again, we get back to a sorted list.

4.2 Commutating

From Exercise 2.1, we saw that in general $\pi_1\pi_2 \neq \pi_2\pi_1$. Nonetheless, if we are swapping disjoint sets of pairs like (12) followed by (34) then there is no interaction between the swaps. So the order doesn't matter: $(12)(34) = (34)(12)$.

```
B.<a,b,c> = BraidGroup(4)
plot(a * c)
```



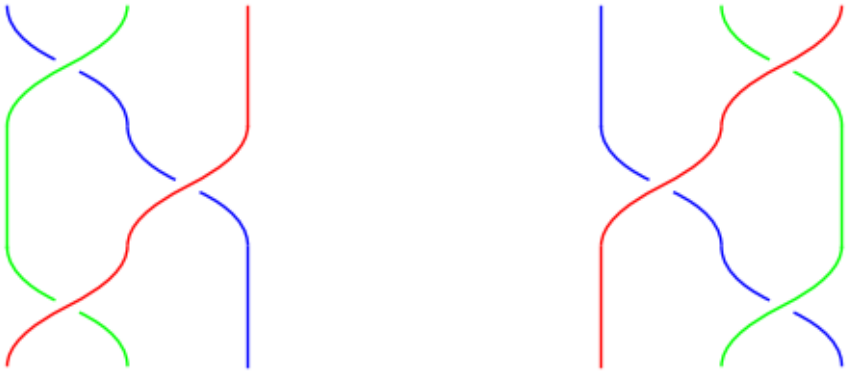
```
a * c == c * a
```

```
True
```

4.3 ABA = BAB

At the top of Section 3.2 we saw that $(12)(23)(12) = (13)$. We also have $(23)(12)(23) = (13)$. Compare the following pictures.

```
B.<a,b> = BraidGroup(3)
plot(a * b * a)
plot(b * a * b)
```



We can see visually that the middle green strand is sliding from one side of the blue/red crossing to the other.