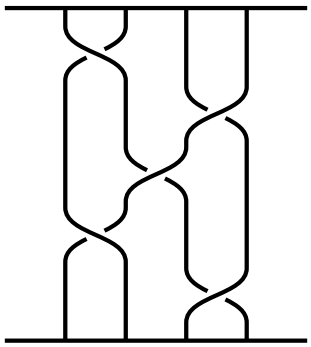
# PROJECT IDEAS

MARCH 22ND 2019

## 1. BRAIDS AND THE BRAID GROUP

A **braid** is a set of $n$ strings, all attached to a horizontal bar at the top and at the bottom, in such a way that the strings can tangle around each other, but must never turn back up: each string travels from the top bar to the bottom bar. An example of a four-strand braid is:



We have many of the same notions for braids as for knots and links (equivalence, Reidemeister moves, triviality...), and we can build knots from braids by "closing up" corresponding strands. In fact, every knot can be constructed in this way! It turns out that braids on $n$ strands have very rich and beautiful algebraic structure, namely they form the braid group $B_n$. In this project, a first goal would be to understand this algebraic structure: how to describe braids from their "smallest pieces", and how these pieces interact. From there, there are many interesting paths this project could take, for example:

- ❉ Learning about a bit of group theory and understanding the group structure of the braid group.
- ❉ Proving that every knot or link is a braid closure.
- ❉ Figuring out when the closure of a braid is an unknot or unlink by understanding Markov's theorem.
- ❉ Determining what the Jones polynomial of any two-strand braid is.

## 2. Generalisations of tricolourability

Recall that a knot (projection) is **tricolourable** if each arc in its projection can be coloured one of three colours satisfying the following conditions:

(1) At least two colours are used in the projection.
(2) At each crossing, either all surrounding arcs have the same colour or are all different colours.

We showed that the trefoil is tricolourable:

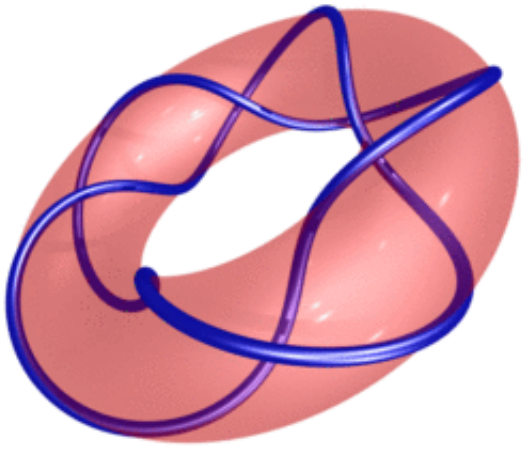
In this project, the aim is to understand some generalisations of tricolourability, in particular:

- Doubled tricolouring: we noted that there are in fact two different trefoils: a left-handed one and a right-handed one. Tricolourability cannot distinguish these, since they are both tricolourable. However, we can define a doubling of this colouring that will in fact distinguish them.
- p-colourability: by labelling arcs by numbers instead of by colours, we can generalise tricolourability to any (prime) number. The aim would be to figure out the relations on the labels required at each crossing, show that the resulting definition of p-colourability is a knot invariant and work out some families of knots that are p-colourable for a chosen value of p.

## 3. TORUS KNOTS

A **torus link** is a link that can be embedded (think "drawn") on a torus (the surface of a donut) without any crossings. These can be described by a pair of integers (p,q): the number of times the link wraps around the torus in the direction of a meridian and around its central "hole". This is an extremely interesting and useful family of knots and links and we can ask ourselves (and answer!) many questions about them:

- Which of our known examples are torus knots?
- When is a torus link in fact a knot?
- What is the relationship between a (p,q)-torus link and a (q,p)-torus link?
- What is the crossing number of a torus link? What is the crossing number of the composition of two torus links?
- What is the Jones polynomial for torus links with small p and q? Can we conjecture a pattern for the polynomial?



## 4. Computing Polynomial Invariants

As we discovered in class, computing polynomial invariants of knots and links such as the Jones polynomial can be very time-consuming and the potential for making mistakes is very high. One way to get around this would be to write a computer program that computes the Jones polynomial for knots and links. In this project, we would discuss the easiest presentation for knots to compute with, some ideas for how to use the skein relation for the Kauffman bracket and how to understand resolutions of crossings.
**Disclaimer:** I cannot teach you how to code, I can only suggest ideas that you would need to implement yourselves in whatever language you already know. This means that this project will only be accessbile to those of you who are very comfortable writing programs like this.

## 5. Seifert surfaces and the genus of a knot

A **Seifert surface** is an oriented surface whose boundary (think: "edge") is a given knot or link. Amazingly, such a surface exists for any knot or link, and Seifert determined an algorithm to compute it. Many knot invariants are defined using Seifert surfaces, so it is a very powerful notion. The main aim of this project will be to understand what the Seifert surface of a knot is and how to compute it, how to define the genus of a knot, then to consider and prove the following satisfying theorems:

- The genus of the composition of two knots is the sum of the genus of knots.
- The unknot is not a composition of knots.
- Genus one knots are prime.

If there is time, we can also make (and hopefully prove) some conjectures on Seifert surfaces with the aid of "SeifertView".

**Note:** This project will require you to do quite a bit of reading to understand what a Seifert surface is, and will introduce you to several notions in topology.