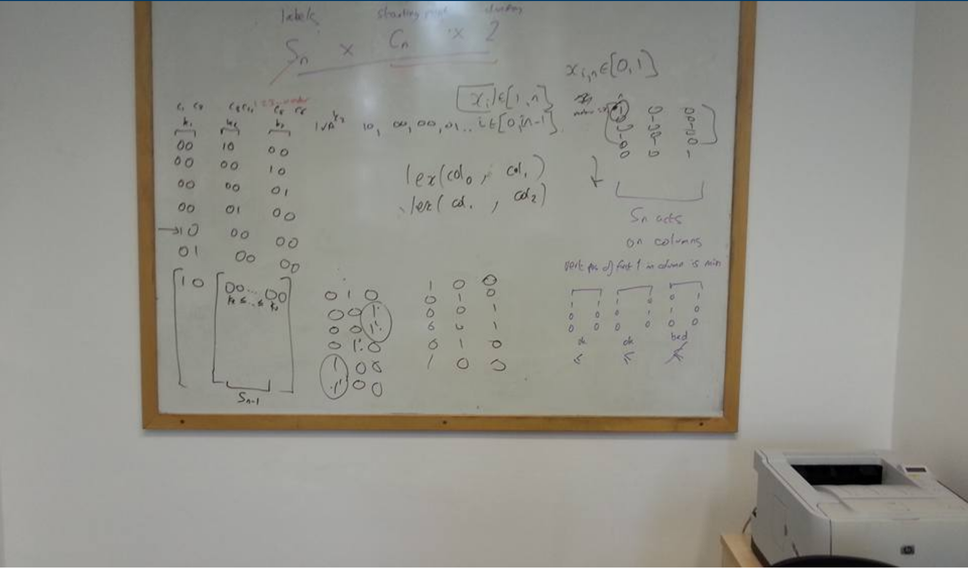
Enumeration of (unique reduced alternating) knot diagrams

Ciaran McCreesh, Alice Miller, Patrick Prosser, **Craig Reilly**, James Trimble



What is a knot?

- A *knot* is an embedding of the circle in $\mathbb{R}^3$.

What is a knot?

- A *knot* is an embedding of the circle in $\mathbb{R}^3$.
- An intuitive way to think about this is to consider a knot as a knotted piece of string with the ends glued together.

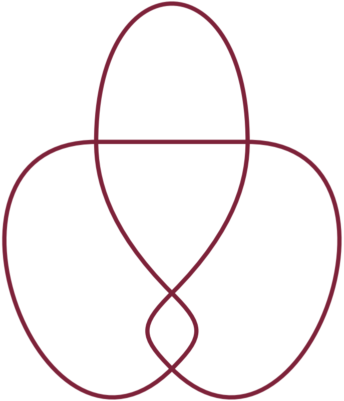
Drawing knots on paper

- A function $f : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ where $f(x, y, z) = f(x, y)$, is called a *projection map*, and the image of a knot K under f is called the *projection* of K.
- Such a projection is often refered to as the *shadow* of K.

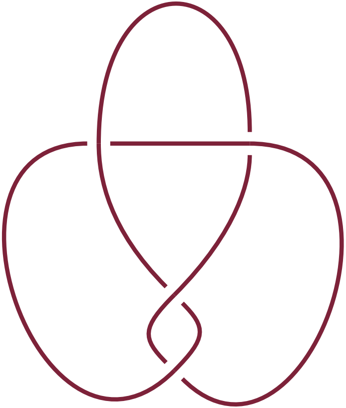
Drawing knots on paper

- Information regarding the orientation of arcs at crossings is given by leaving gaps in a knot's shadow.

Drawing knots on paper



Drawing knots on paper

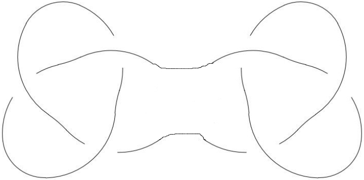


Prime knots

- *Prime knots* are knots which cannot be decomposed. For nontrivial knots this means that they cannot be written as the connect sum of two non-trivial knots.

Prime knots

- *Prime knots* are knots which cannot be decomposed. For nontrivial knots this means that they cannot be written as the connect sum of two non-trivial knots.



Equivelent knots and Reidemeister Moves

- When enumerating prime knots it is convention to give each prime knot by a knot diagram which has as few as possible crossings.
- Two knots K and J are equivalent if K can be transformed into J by a series of elementary deformations.
- Showing that two knots are equivalent by elementary deformations is not practical.

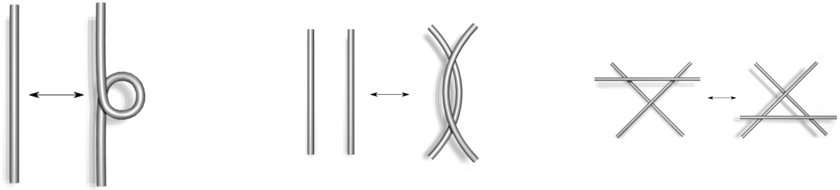
Equivelent knots and Reidemeister Moves

- In the 1926 Reidermister proved that two knot diagrams k_0 and k_1 of the same knot K can be related by a sequence of the following three moves (called Reidermeister moves):[1]

[1] Alexander and Briggs developed these moves independently in 1927

Equivelent knots and Reidemeister Moves

- In the 1926 Reidermister proved that two knot diagrams k_0 and k_1 of the same knot K can be related by a sequence of the following three moves (called Reidermeister moves):[1]



[1] Alexander and Briggs developed these moves independently in 1927

Representations of knots

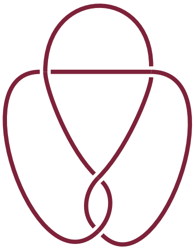
- Knot diagrams are really just 4-valent planar graphs.
  - The vertices in the graph correspond to the crossings in the knot diagram.
  - The arcs between vertices correspond to arcs between crossings in the knot diagram.
  - The arcs are decorated with their orientation at their source and target crossings.
- Other data structures familiar to computer scientists can be used, linked lists of crossings were popular in the 1950's.

Representations of knots

- The representations used by topologists are typically also used for representing knots in a computer.
- Examples are Dowker-Thistlethwait codes (DT codes), **Gauss codes**, braid representatives, Conway notation, and many more.

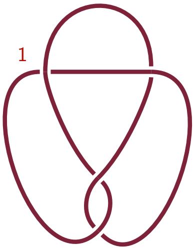
Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



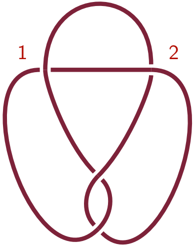
Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



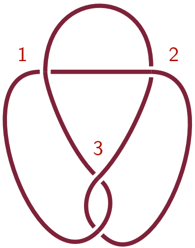
Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



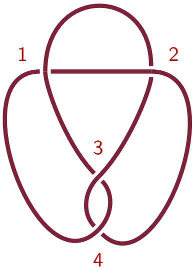
Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



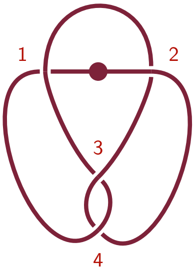
Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



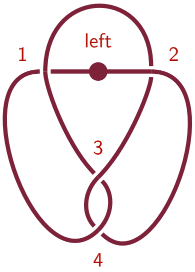
Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



Representing knots with Gauss codes

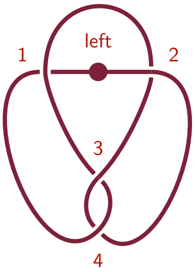
- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



Representing knots with Gauss codes

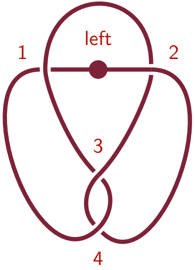
- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.

-1



Representing knots with Gauss codes

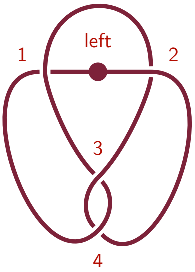
- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.



-1 , 4

Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.

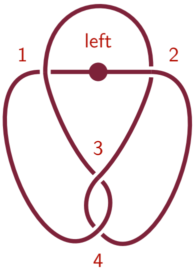


$-1 \;\;, 4 \;\;, -3$

Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.

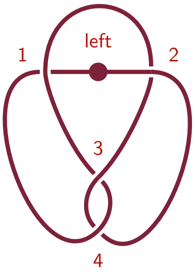
$-1\ ,\ 4\ ,\ -3\ ,\ 1$



Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.

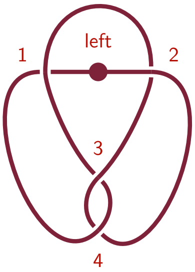
$-1\ ,\ 4\ ,\ -3\ ,\ 1\ ,\ -2$



Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.

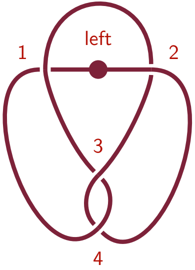
$-1 \;, 4 \;, -3 \;, 1 \;, -2 \;, 3$



Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.

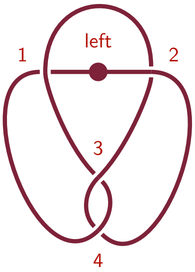
$-1\ ,\ 4\ ,\ -3\ ,\ 1\ ,\ -2\ ,\ 3\ ,\ -4$



Representing knots with Gauss codes

- The strategy for representing a given knot (with n crossings) by a Gauss code is as follows.
  1. Label the crossings with the numbers 1 to n.
  2. Pick a point on the knot.
  3. Pick a direction and walk around the knot, writing out a list of the numbers you come to (with a negative sign indicating that a crossing was visited on an under strand). Stop when each number appears twice once with each sign.

-1 , 4 , -3 , 1 , -2 , 3 , -4 , 2



Knot enumeration by Gauss codes

- There are three sensible questions to ask:
  - Do all knots give rise to a Gauss code?
  - Can two Gauss codes correspond to the same knot?
  - Do all Gauss codes represent a knot?

Knot enumeration by Gauss codes

- There are three sensible questions to ask:
  - Do all knots give rise to a Gauss code? Yes.
  - Can two Gauss codes correspond to the same knot?
  - Do all Gauss codes represent a knot?

Knot enumeration by Gauss codes

- There are three sensible questions to ask:
  - Do all knots give rise to a Gauss code? Yes. Trivially.
  - Can two Gauss codes correspond to the same knot?
  - Do all Gauss codes represent a knot?

Knot enumeration by Gauss codes

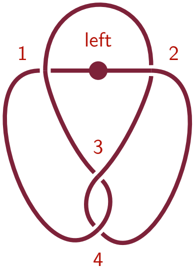
- There are three sensible questions to ask:
  - Do all knots give rise to a Gauss code? Yes. Trivially.
  - Can two Gauss codes correspond to the same knot? Yes.
  - Do all Gauss codes represent a knot?

Knot enumeration by Gauss codes

- There are three sensible questions to ask:
  - Do all knots give rise to a Gauss code? Yes. Trivially.
  - Can two Gauss codes correspond to the same knot? Yes.
  - Do all Gauss codes represent a knot? No!

A candidate for a canonical Gauss code

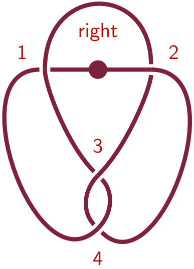
$-1, 4, -3, 1, -2, 3, -4, 2$



A candidate for a canonical Gauss code

$-1, 4, -3, 1, -2, 3, -4, 2$

$2, -4, 3, -2, 1, -3, 4, -1$

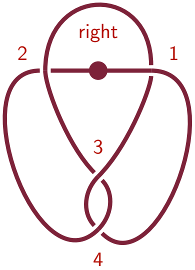


A candidate for a canonical Gauss code

$-1, 4, -3, 1, -2, 3, -4, 2$

$2, -4, 3, -2, 1, -3, 4, -1$

$1, -4, 3, -1, 2, -3, 4, -2$



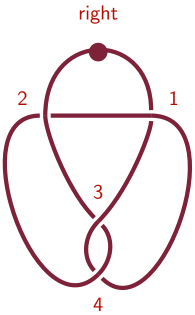
A candidate for a canonical Gauss code

$-1, 4, -3, 1, -2, 3, -4, 2$

$2, -4, 3, -2, 1, -3, 4, -1$

$1, -4, 3, -1, 2, -3, 4, -2$

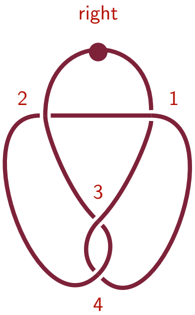
$-1, 3, -4, 1, -2, 4, -3, 2$



A candidate for a canonical Gauss code

- Our candidate is a code with the lexicographically minimum shadow (where a shadow corresponds to an unsigned code). For this knot the shadow is:

  1, 2, 3, 1, 4, 3, 2, 4.



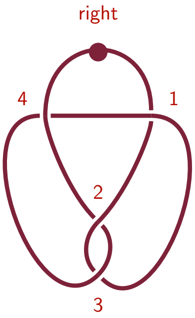
A candidate for a canonical Gauss code

- Our candidate is a code with the lexicographically minimum shadow (where a shadow corresponds to an unsigned code). For this knot the shadow is:

  1, 2, 3, 1, 4, 3, 2, 4.

- For which the diagram now shows the labelling giving rise to the signed code:

  -1, 2, -3, 1, -4, 3, -2, 4.



A candidate for a canonical Gauss code

- Is this new?

A candidate for a canonical Gauss code

- Is this new? Kinda.

A candidate for a canonical Gauss code

- Is this new? Kinda.
- Topologists don't seem to care about labelling a knot in a canonical manner when determining a knot's Gauss code.
- But such a lexicographically minium representation of the unsigned Gauss code is necessary input of an algorithm which follows.

A candidate for a canonical Gauss code

Name	Gauss Notation
3_1	−1, 3, −2, 1, −3, 2
4_1	1, −4, 3, −1, 2, −3, 4, −2
5_1	−1, 4, −2, 5, −3, 1, −4, 2, −5, 3
5_2	−1, 5, −2, 1, −3, 4, −5, 2, −4, 3
6_1	−1, 4, −3, 1, −5, 6, −2, 3, −4, 2, −6, 5
6_2	−1, 4, −3, 1, −2, 6, −5, 3, −4, 2, −6, 5
6_3	1, −6, 2, −1, 4, −5, 6, −2, 3, −4, 5, −3

Symmetries involved in Gauss code enumeration

- The choices involved in each stage of obtaining a Gauss code from a given knot lead to rich symmetries in the problem of enumerating knots by Gauss codes.
- As we have seen many Gauss codes represent the same knot.

Symmetries involved in Gauss code enumeration

Symmetries involved in Gauss code enumeration

- The group of the symmetries of the problem is $S_{2n} \times Z_{2n} \times Z_2$ (over some quotient).
- Symmetry breaking on each of these groups is easy, but this is not the case for their product.

Virtual knots

Virtual knots

- The field of virtual knot theory arose from the question "do all Gauss codes represent knots?"
- Louis Kauffman lead the research into this new field and published his introduction to the subject in 1998.
- This introduction includes an algorithm for determining if a knot is classical or virtual.

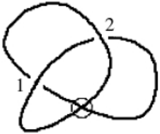
Virtual knots

- It isn't hard to think of an example of a Gauss code which does not represent a classical knot.

Virtual knots

- It isn't hard to think of an example of a Gauss code which does not represent a classical knot.

  $1, -2, -1, 2$



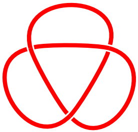
Evenly spaced Gauss codes

- Kauffman shows that a necessary condition for Gauss codes to give classical knots is that they are evenly spaced.

Evenly spaced Gauss codes

- Kauffman shows that a necessary condition for Gauss codes to give classical knots is that they are evenly spaced.

  1, 2, 3, 1, 2, 3

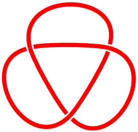


Evenly spaced Gauss codes

- Kauffman shows that a necessary condition for Gauss codes to give classical knots is that they are evenly spaced.

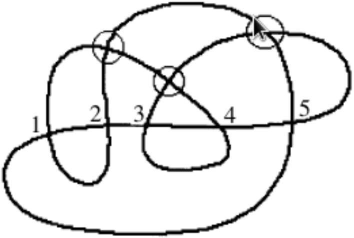
  1, 2, 3, 1, 2, 3

  1, 2, 3, 1, 4, 3, 2, 4



Evenly spaced Gauss codes

- However, he also provides an example where this condition is not sufficient.



1234534125

The dually pairedness condition

- The other necessary condition, which when combined with the evenly spaced condition is sufficient, is that, for a Gauss code w its associated w^* must be dually paired.

The dually pairedness condition

- The other necessary condition, which when combined with the evenly spaced condition is sufficient, is that, for a Gauss code w its associated w^* must be dually paired.
- For a Gauss code w, its associated w^* is obtained by the following algorithm: for each letter i in the code, reverse the order of all the letters between each instance of i in turn.

The dually pairedness condition

Original code 1, 2, 3, 1, 4, 3, 2, 4

Order between 1 reversed 1, 3, 2, 1, 4, 3, 2, 4

The dually pairedness condition

Original code 1, 2, 3, 1, 4, 3, 2, 4

Order between 1 reversed 1, 3, 2, 1, 4, 3, 2, 4

Order between 2 reversed 1, 3, 2, 3, 4, 1, 2, 4

The dually pairedness condition

Original code 1, 2, 3, 1, 4, 3, 2, 4

Order between 1 reversed 1, 3, 2, 1, 4, 3, 2, 4

Order between 2 reversed 1, 3, 2, 3, 4, 1, 2, 4

Order between 3 reversed 1, 3, 2, 3, 4, 1, 2, 4

The dually pairedness condition

Original code 1, 2, 3, 1, 4, 3, 2, 4

Order between 1 reversed 1, 3, 2, 1, 4, 3, 2, 4

Order between 2 reversed 1, 3, 2, 3, 4, 1, 2, 4

Order between 3 reversed 1, 3, 2, 3, 4, 1, 2, 4

Order between 4 reversed 1, 3, 2, 3, 4, 2, 1, 4

The dually pairedness condition

- For a shadow Gauss code w two labels i and j are called *interlaced* in w iff $w = w_1\ i\ w_2\ j\ w_3\ i\ w_4\ j\ w_5$ or $w = w_1\ j\ w_2\ i\ w_3\ j\ w_4\ i\ w_5$, where each w_n is a subcode (which is possibly empty).
- An unsigned Gauss code's w^* construction is *dually paired* iff each letter $i \in w^*$ can be placed into one of two sets, such that each i in both sets is not interlaced with another letter contained in the same set as i.

The dually pairedness condition

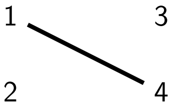
- For a shadow Gauss code w two labels i and j are called *interlaced* in w iff $w = w_1\ i\ w_2\ j\ w_3\ i\ w_4\ j\ w_5$ or $w = w_1\ j\ w_2\ i\ w_3\ j\ w_4\ i\ w_5$, where each w_n is a subcode (which is possibly empty).
- An unsigned Gauss code's w^* construction is *dually paired* iff each letter $i \in w^*$ can be placed into one of two sets, such that each i in both sets is not interlaced with another letter contained in the same set as i.

1 3

2 4

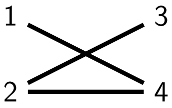
The dually pairedness condition

- For a shadow Gauss code w two labels i and j are called *interlaced* in w iff $w = w_1\ i\ w_2\ j\ w_3\ i\ w_4\ j\ w_5$ or $w = w_1\ j\ w_2\ i\ w_3\ j\ w_4\ i\ w_5$, where each w_n is a subcode (which is possibly empty).
- An unsigned Gauss code's w^* construction is *dually paired* iff each letter $i \in w^*$ can be placed into one of two sets, such that each i in both sets is not interlaced with another letter contained in the same set as i.



The dually pairedness condition

- For a shadow Gauss code w two labels i and j are called *interlaced* in w iff $w = w_1\ i\ w_2\ j\ w_3\ i\ w_4\ j\ w_5$ or $w = w_1\ j\ w_2\ i\ w_3\ j\ w_4\ i\ w_5$, where each w_n is a subcode (which is possibly empty).
- An unsigned Gauss code's w^* construction is *dually paired* iff each letter $i \in w^*$ can be placed into one of two sets, such that each i in both sets is not interlaced with another letter contained in the same set as i.



Kauffman's algorithm

```
isPlanarUnsignedGaussCode (int[] gc) → Bool
begin
    if gc is not evenly spaced then return false
    else gcStar ← createWStar(gc)
    return whether gcStar is not dually paired
```

Algorithm 1: Kauffman's algorithm for recognising planar unsigned Gauss codes

Implementation

- The aim of the project was to generate all knot diagrams of a given crossings number, up to some notion of equivalence.
- The project made use of both Constraint Programming and dedicated algorithms to efficently filter codes.

Implementation

- Constraint programming:
  - Generate Gauss codes which pass the evenly spaced condition.
  - Symmetry breaking on the S_n symmetry.
- Filtering:
  - Test that the code is dually paired.
  - Test that the code is the lexicographically minium code in its orbit in the group $\mathbb{Z}_{2n} \times \mathbb{Z}_2$.

A naive constraint model

For a knot with crossing number n:

$$\forall i \in \{0, 2n{-}1\} : \mathit{letter}[i] \in \{1, n\} \tag{V1}$$

$$\forall k \in \{1, n\} : \mathrm{occurrence}(\mathit{letter}[i] = k) = 2 \tag{C1}$$

$$\forall i, j \in \{0, 2n{-}1\}, i \neq j, \text{ with } i \text{ even } : \mathit{letter}[i] = k \implies \mathit{letter}[j] \neq k \text{ for } j \text{ even} \tag{C2}$$

$$\forall i, j \in \{0, 2n{-}1\}, i \neq j, \text{ with } i \text{ odd } : \mathit{letter}[i] = k \implies \mathit{letter}[j] \neq k \text{ for } j \text{ odd} \tag{C3}$$

$$\forall i \in \{0, 2n{-}1\} : \mathit{maxSoFar}[i] = \max(\mathit{letter}[0], \ldots, \mathit{letter}[i-1]) \tag{V2}$$

$$\forall i \in \{0, 2n{-}1\} : \mathit{letter}[i] \leq \mathit{maxSoFar}[i] + 1 \tag{C4}$$

$$\mathit{letter}[0] = 1 \tag{C5}$$

$$\forall i \in \{n, 2n-1\} : \mathit{letter}[i] \neq 1 \tag{C6}$$

A model using 0/1 variables

For a knot with crossing number n:

$$\forall i \in \{0, 2n{-}1\}, \forall j \in \{0, n-1\} : \mathit{letter}[i][j] \in \{0,1\} \tag{V3}$$
$$\forall i \in \{0, n{-}1\}, \forall j \in \{0, 2n-1\} : \mathit{letterTranspose}[i][j] \in \{0,1\}$$
$$\forall i \in \{0, n{-}1\}, \forall j \in \{0, n{-}1\} : \mathit{evenLetter}[i][j] \in \{0,1\}$$
$$\forall i \in \{0, n{-}1\}, \forall j \in \{0, n{-}1\} : \mathit{evenLetterTranspose}[i][j] \in \{0,1\}$$
$$\forall i \in \{0, n{-}1\}, \forall j \in \{0, n{-}1\} : \mathit{oddLetter}[i][j] \in \{0,1\}$$
$$\forall i \in \{0, n{-}1\}, \forall j \in \{0, n{-}1\} : \mathit{oddLetterTranspose}[i][j] \in \{0,1\}$$

A model using 0/1 variables

$$
\begin{aligned}
&\forall i \in \{0, 2n{-}1\}, \forall j \in \{0, n-1\} : \\
&\qquad \mathit{letterTranspose}[i][j] = \mathit{letter}[j][i] \\
&\qquad \mathit{evenLetter}[i][j] = \mathit{letter}[i][j] \text{ for } i \text{ even} \\
&\qquad \mathit{evenLetterTranspose}[i][j] = \mathit{evenLetter}[j][i] \\
&\qquad \mathit{oddLetter}[i][j] = \mathit{letter}[i][j] \text{ for } i \text{ odd} \\
&\qquad \mathit{oddLetterTranspose}[i][j] = \mathit{oddLetter}[j][i]
\end{aligned}
$$

A model using 0/1 variables

$$\forall i \in \{0, 2n{-}1\} : \sum(\mathit{letter}[i]) = 1 \tag{C7}$$

$$\forall i \in \{0, n{-}1\} : \sum(\mathit{letterTranspose}[i]) = 2 \tag{C8}$$

$$\forall i \in \{0, n{-}1\} : \sum(\mathit{evenLetterTranspose}[i]) = 1 \tag{C9}$$

$$\forall i \in \{0, n{-}1\} : \sum(\mathit{oddLetterTranspose}[i]) = 1 \tag{C10}$$

$$\forall i \in \{1, n{-}1\} : \mathrm{lex}(\mathit{letterTranspose}[i], \mathit{letterTranspose}[i+1]) \tag{C11}$$

$$\mathit{letter}[0][0] = 1 \tag{C12}$$

The trefoil represented by the 0/1 model

1	0	0
0	1	0
0	0	1
1	0	0
0	1	0
0	0	1

Further work

- Add constraints to the model to only admit knots which aren't connect sums of two knots, and don't have and Reidermeister 1 moves.
- Parallelise the enumeration.