Hard problems in Knot Theory

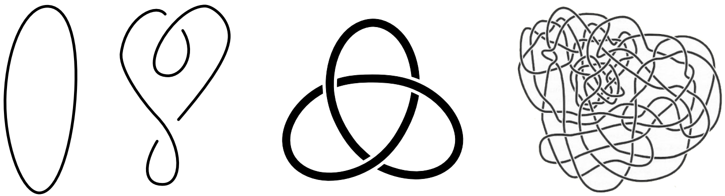
Arnaud de Mesmay (CNRS, GIPSA-lab)

Joint work with Yo'av Rieck, Eric Sedgwick and Martin Tancer.

Knot Theory

Knots

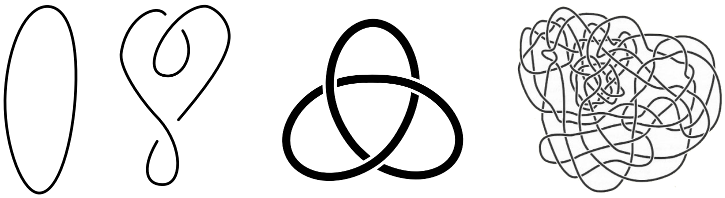
- A *knot* is a nice map $K : S^1 \to \mathbb{R}^3$.
- Two knots are *equivalent* if they are *ambient isotopic*, i.e., if there exists a continuous deformation from one into the other without crossings.
- It is not obvious that there exist non-equivalent knots.



Knot diagrams

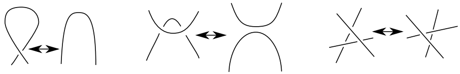
Diagrams

- A *knot diagram* is a 2D-projection of a knot where at every vertex, one indicates which strand goes above and below.



Theorem (Reidemeister)

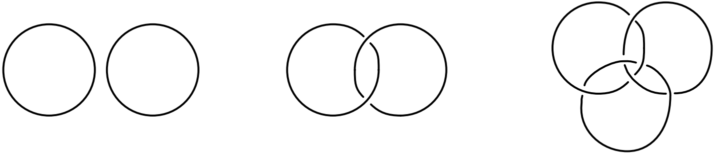
Two knot diagrams correspond to equivalent knots if and only if they can be related by a sequence of Reidemeister moves.



Links

Links

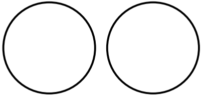
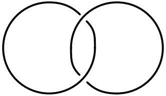
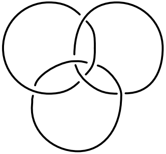
- A *link* is a disjoint union of knots.
- Two links are equivalent if they are ambient isotopic.

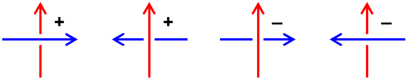


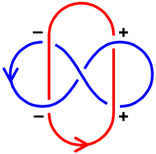
Links

Links

- A *link* is a disjoint union of knots.
- Two links are equivalent if they are ambient isotopic.
- It is easy to prove that there exists non equivalent links by using the *linking number*...
- ... but it does not work all the time

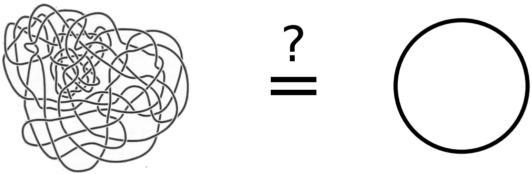




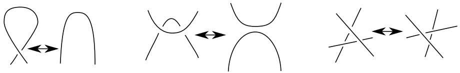
Computational problems in knot theory I

Unknot recognition

Input: A piecewise-linear closed curve K in $\mathbb{R}^3$ made of n segments.
Output: Is K equivalent to the trivial knot?



- Already not obvious that this is decidable [Haken '61].
- In $\mathbf{NP} \cap co-\mathbf{NP}$ [Hass-Lagarias-Pippenger '99], [Agol'02 $\rightarrow$ Lackenby '18].
- "Easy" **NP** algorithm: guess a sequence of Reidemeister moves. $O(n^{11})$ moves are enough [Lackenby'15].

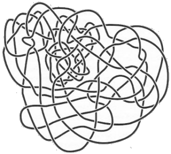


- Lower bounds: ??

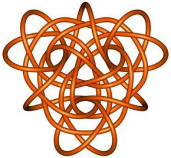
Computational problems in knot theory II

Knot equivalence

Input: Two closed curves K_1 and K_2 in $\mathbb{R}^3$ made of n_1 and n_2 segments.
Output: Is K_1 equivalent to K_2?



$\stackrel{?}{=}$



- Even harder to prove that it is decidable [Hemion '79, Matveev '07].
- Best known algorithm [Lackenby-Coward '14], via Reidemeister moves:

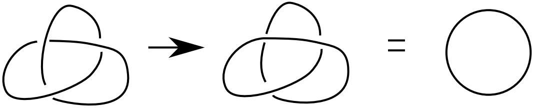
$$\left. 2^{2^{\cdot^{\cdot^{\cdot^{2^{n_1+n_2}}}}}} \right\} \text{height } c^{n_1+n_2} \text{ where } c = 10^{1000000}.$$

- Lower bounds: ???

Computational problems in knot theory III

Unknotting number

Input: A closed curve K in $\mathbb{R}^3$ made of n segments and an integer k.
Output: Can I make k crossing changes to K to transform it into a trivial knot?



- Crossing changes are allowed in *any* diagram of K.
- Not known to be decidable.
- No known lower bound.
- Same for the *Unlinking number*: turning a link into a trivial link.

Our results

Finding the best way to untangle a knot is hard:

Theorem

Deciding whether there is a sequence of at most k Reidemeister moves transforming a knot diagram into the unknot is ***NP****-hard.*

Finding the best way to cut a link to untangle it is hard:

Theorem

Computing the unlinking number of a link is ***NP****-hard.*

Finding an untangled sublink is hard:

Theorem

Determining if a link admits a trivial sublink with n components is ***NP****-hard.*

- We also get hardness for a host of 4-dimensional invariants.
- Similar simultaneous results/conjecture in [Koenig,Tsvletkova'18].
- Only two (!) hard problems known for knots.

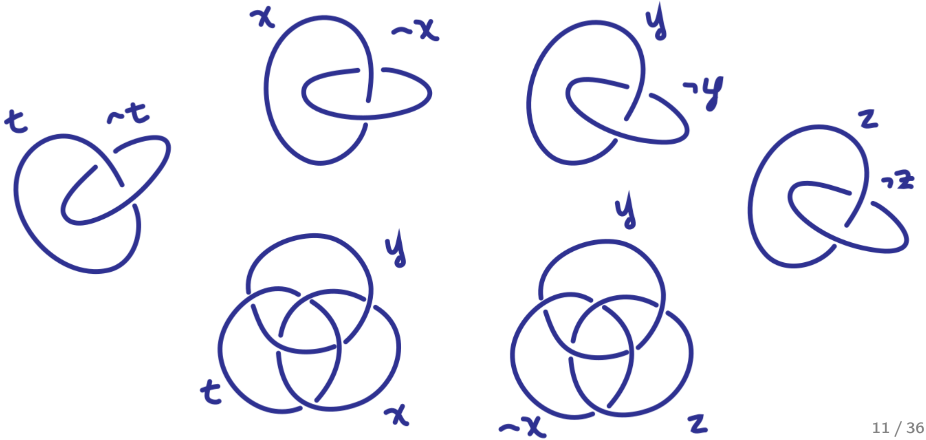
Trivial sublink

Input: A link L made of n segments, an integer k.
Output: Does L contain a trivial sublink with k components?

Trivial sublink

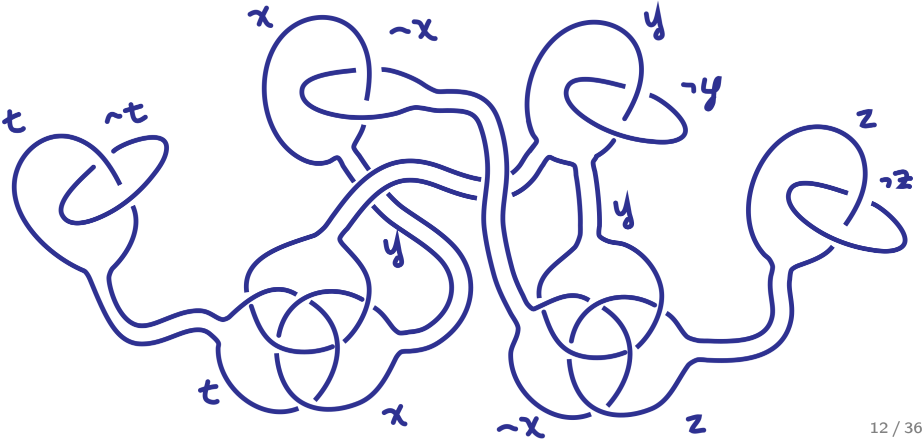
Input: A link L made of n segments, an integer k.
Output: Does L contain a trivial sublink with k components?

$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Input: A link L made of n segments, an integer k.
Output: Does L contain a trivial sublink with k components?

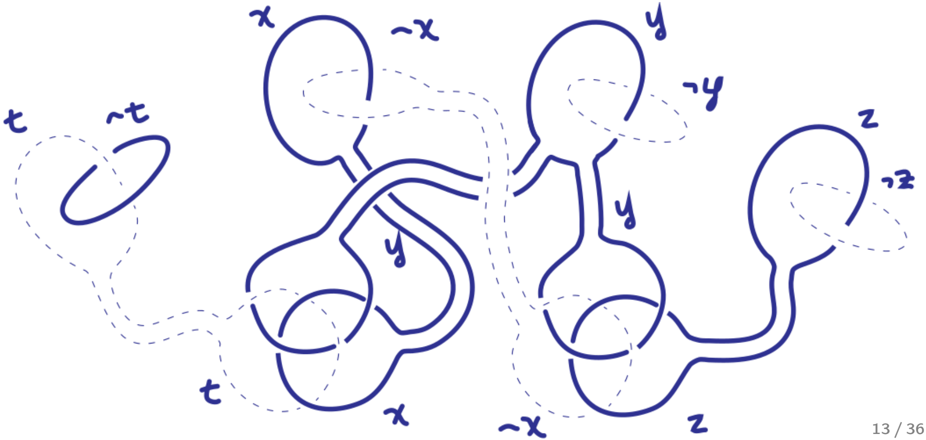
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Trivial sublink

First direction: Remove the components assigned as true.
The rest is unlinked!

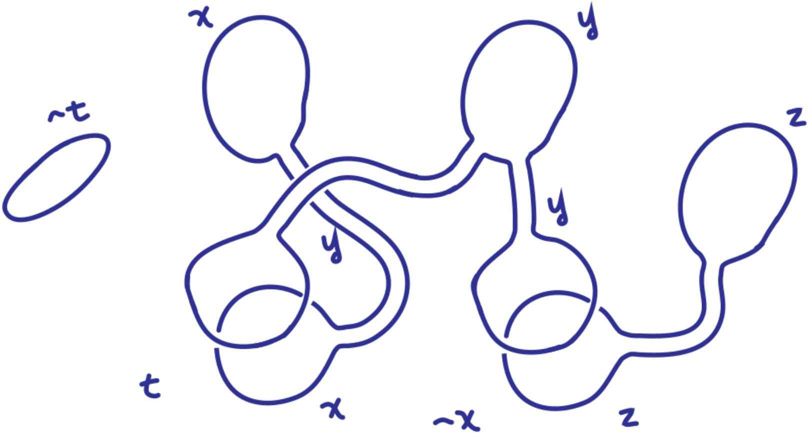
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Trivial sublink

First direction: Remove the components assigned as true.
The rest is unlinked!

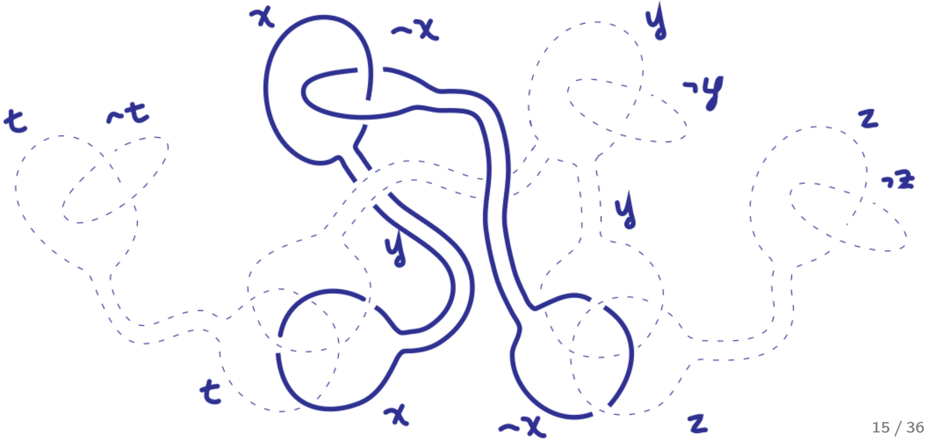
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Trivial sublink

Second direction: For each variable, at least one literal has been removed. For each clause, at least one literal has been removed.

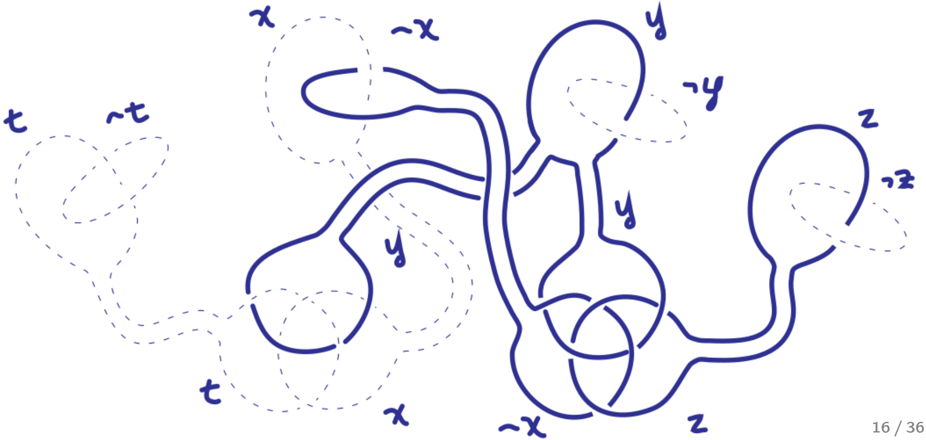
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Trivial sublink

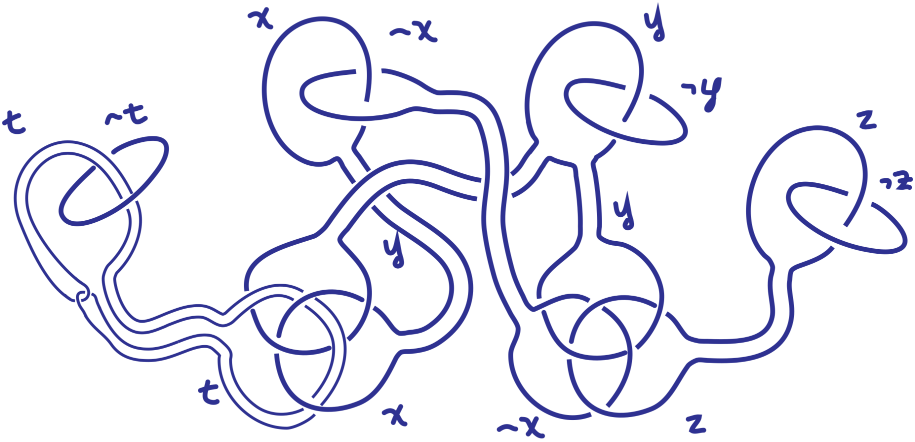
Second direction: For each variable, at least one literal has been removed. For each clause, at least one literal has been removed.

$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Unlinking number

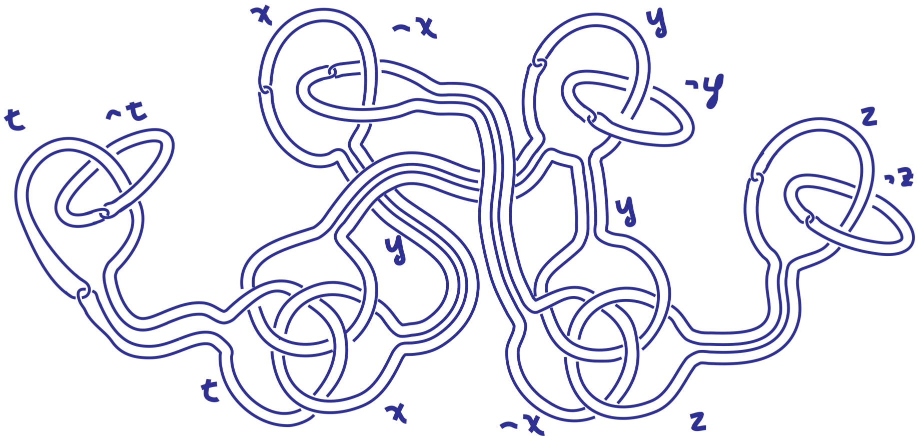
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Replace each knot by its *Whitehead double*.

Unlinking number

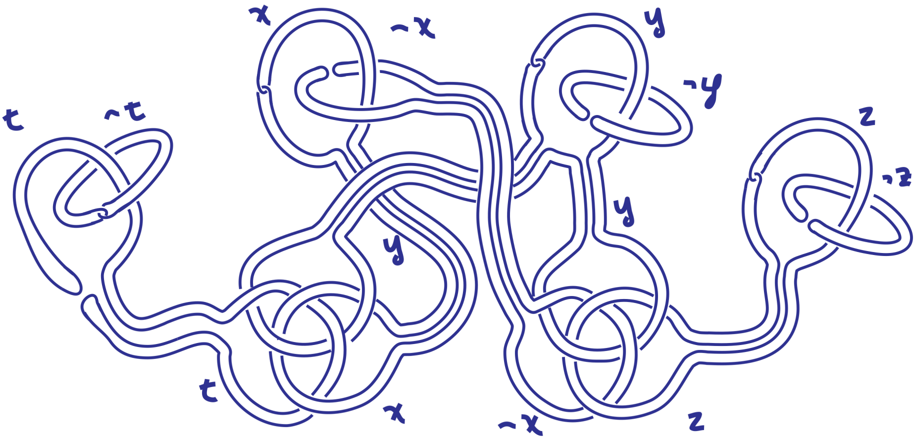
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Replace each knot by its *Whitehead double*.

Unlinking number

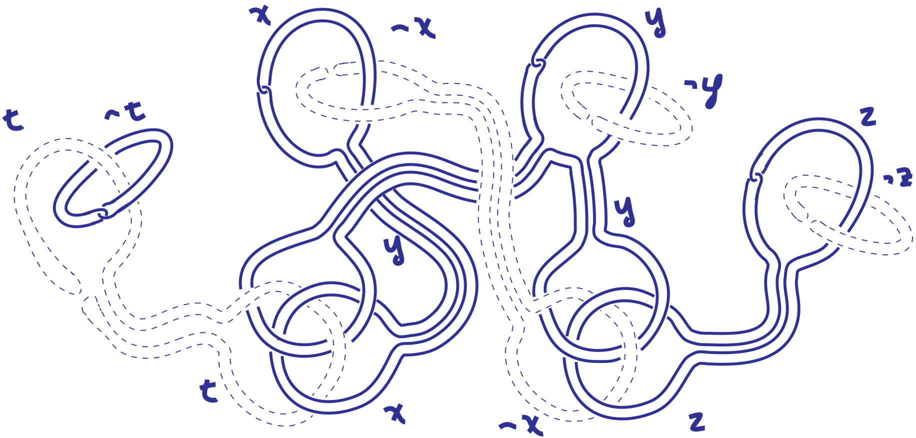
$$\phi = (t \lor x \lor y) \land (\neg x \lor y \lor z)$$



If Φ is satisfiable, uncross the true literals (n crossing changes).

Unlinking number

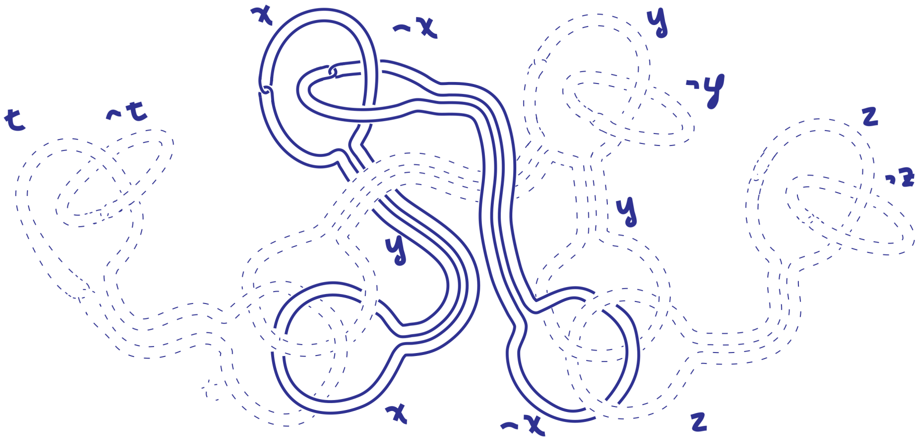
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



We get a trivial link.

Unlinking number

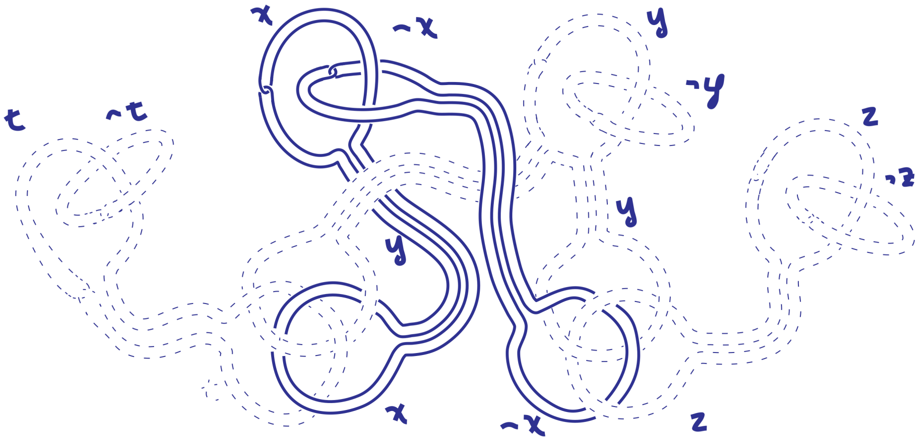
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Unlinkable in n changes $\rightarrow$ each variable gets a crossing change.

Unlinking number

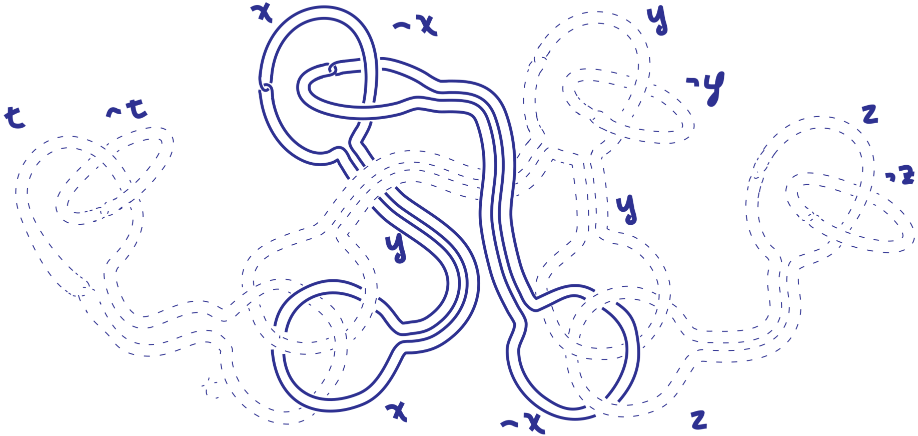
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



This crossing change hits x or $\neg x$ but not both.

Unlinking number

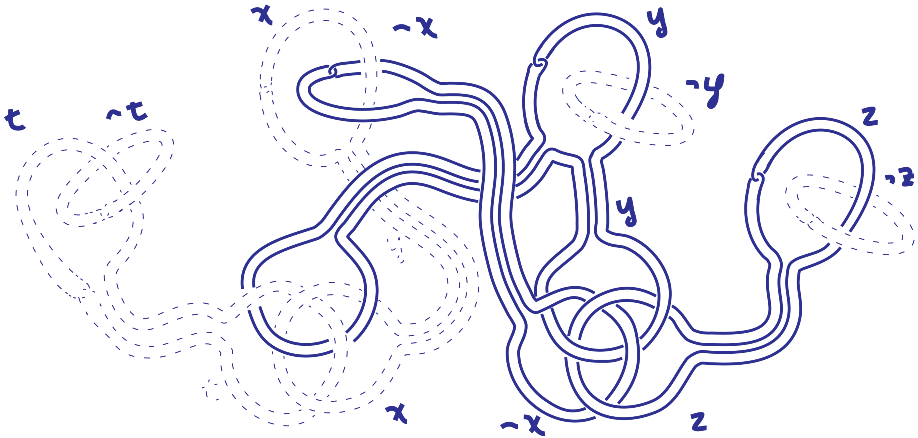
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



We call the corresponding literal TRUE.

Unlinking number

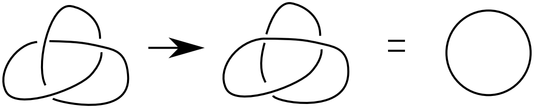
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Each clause (Borromean rings) sees at least one crossing change $\rightarrow$ Φ satisfiable.

Parenthesis on crossing changes

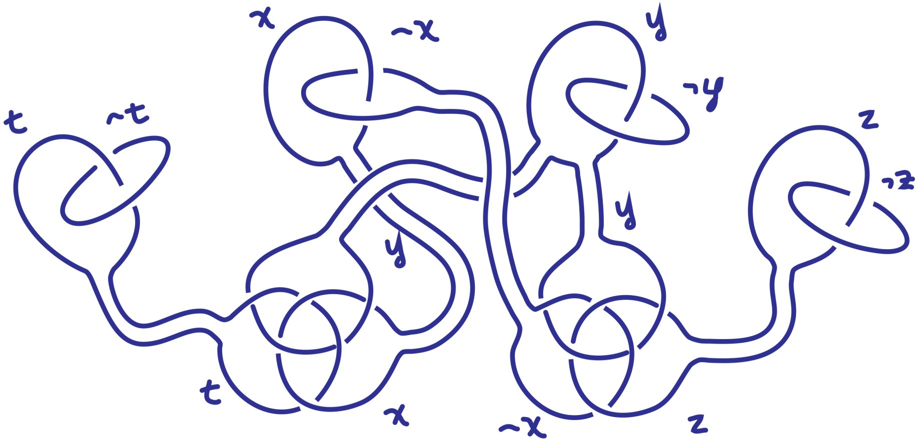
- Embedding into $\mathbb{R}^3$ or $\mathbb{S}^3$ is equivalent.
- $\mathbb{S}^3$ is the boundary of $\mathbb{B}^4$.
- The unlinking/unknotting number is related to the genus of surfaces in $\mathbb{B}^4$ having the knot as a boundary.
- Making a crossing change amounts to adding a handle to the surface.



$\Rightarrow$ Hardness of a handful of 4-dimensional invariants.

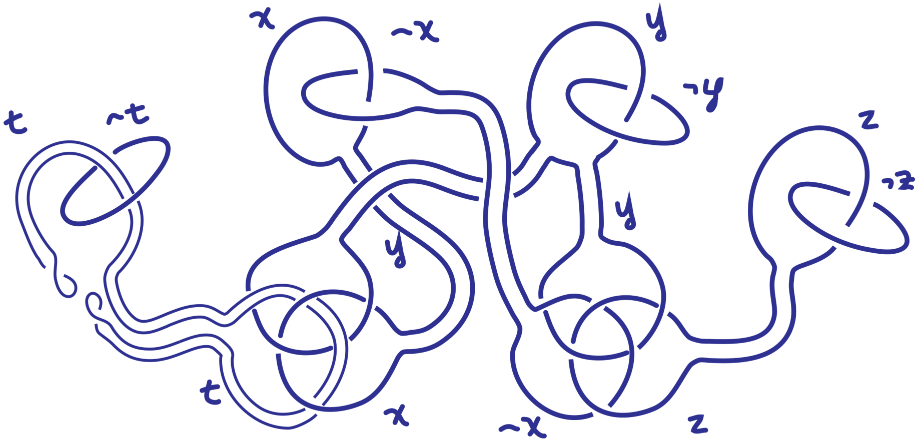
Reidemeister moves I

$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Reidemeister moves I

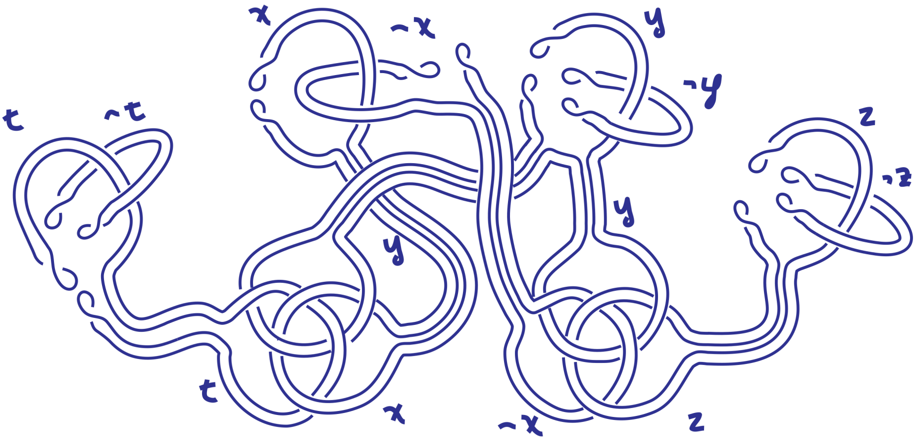
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Replace each knot with a twisted trivial knot.

Reidemeister moves I

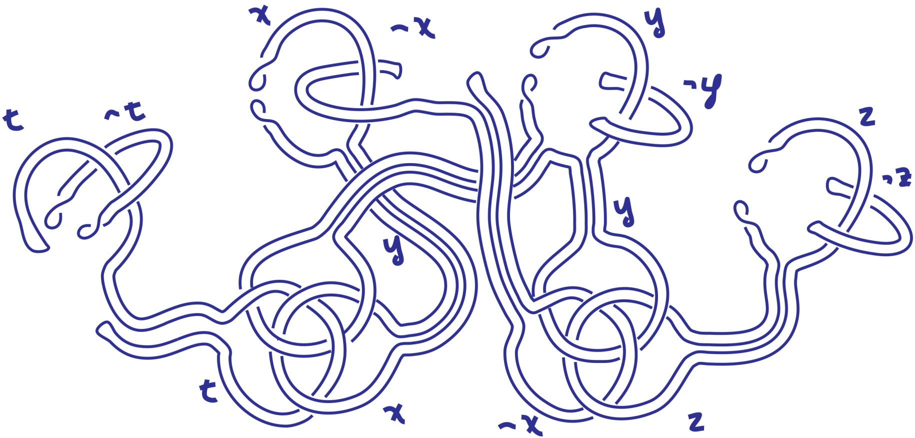
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



We get the diagram of a trivial link.

Reidemeister moves I

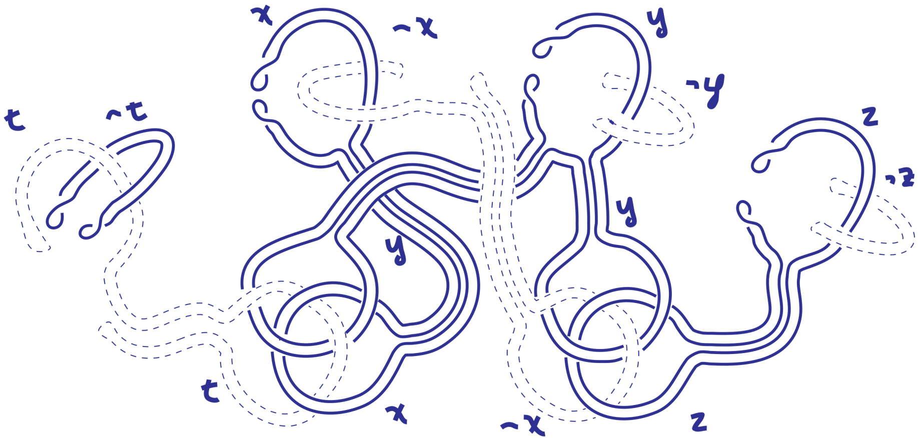
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



Untwist the ends of TRUE components ($2n$ Reidemeister I).

Reidemeister moves I

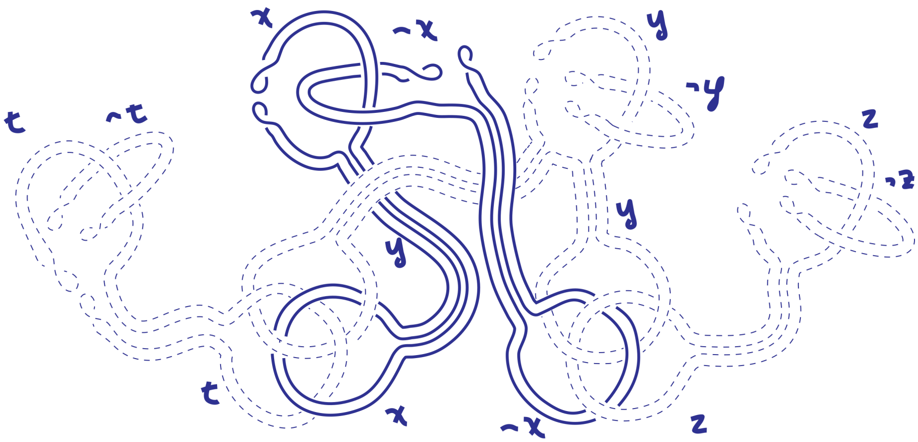
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



The rest unravels with Reidemeister II.

Reidemeister moves I

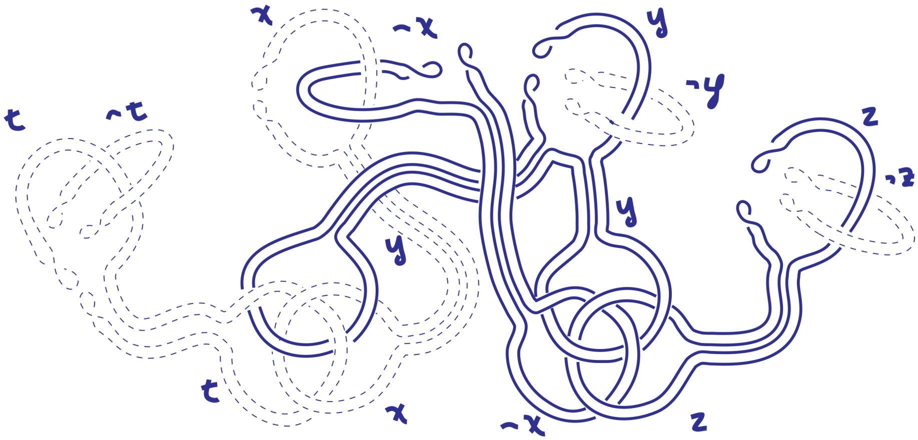
$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$



In the other direction, look separately at variables...

Reidemeister moves I

$$\phi = (t \vee x \vee y) \wedge (\neg x \vee y \vee z)$$

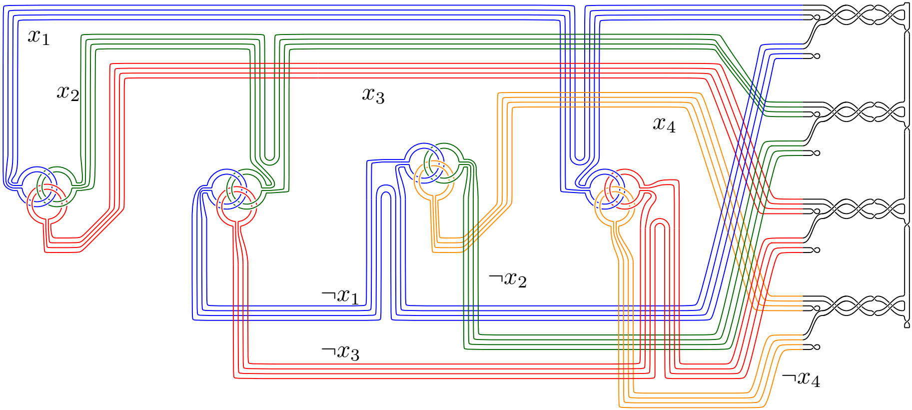


... and clauses.

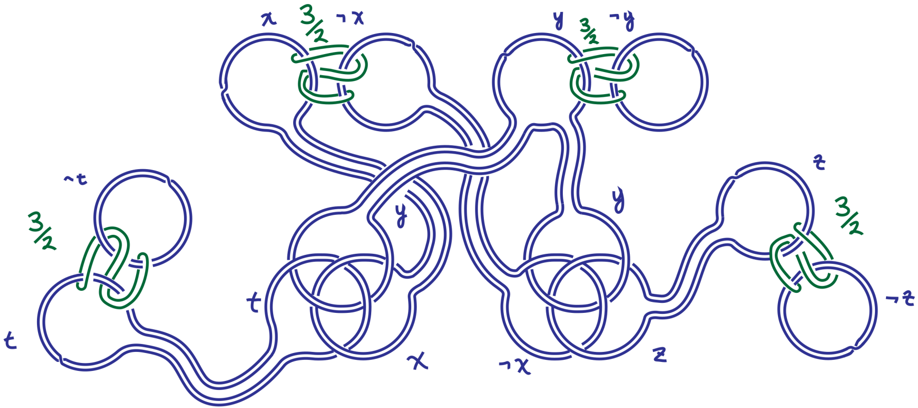
Reidemeister moves II

With (a lot of) additional work, we can deal with the case of a single knot.

$$\Phi = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4)$$



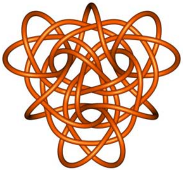
Our original motivation: Embed$_{2\to3}$ and Embed$_{3\to3}$



This 3-manifold embeds into $\mathbb{S}^3$ if and only if Φ is satisfiable.
$\rightarrow$ Deciding whether a 3 or a 2-dimensional space embeds into $\mathbb{R}^3$ is **NP**-hard.
Best algorithm runs in a tower of exponentials [Matoušek, Sedgwick, Tancer, Wagner '16].

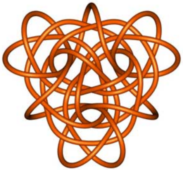
Perspectives

- Unknotting number?
- Knot equivalence?
- Crossing number (= smallest number of crossings in a diagram)? The best known algorithm is the naive one.



Perspectives

- Unknotting number?
- Knot equivalence?
- Crossing number (= smallest number of crossings in a diagram)? The best known algorithm is the naive one.



Thank you! Questions?