Relative Strengths of Knot Invariants by Experiment

Ryan Maguire

April 1, 2024

Outline

1. The Alexander Polynomial.
2. The Jones Polynomial.
3. Khovanov Homology.
4. The HOMFLY-PT Polynomial.
5. Results and Statistics.

The Alexander Polynomial

One of the oldest invariants in knot theory, the Alexander polynomial is not-so-easy to describe, but very easy to compute. The original definition involves a certain infinite cyclic cover of the complement of a knot embedded into $\mathbb{R}^3$ (see [Lic97, p. 53]), but the computation can be explained fairly quickly using knot diagrams.

For a planar diagram of a knot there will be N crossings and $N+2$ faces using Euler's formula. Create a $N \times (N+2)$ adjacency matrix and fill the entries with affine combinations of terms in the set $\{1, -1, t, -t\}$ based on the orientation of the crossings. [Liv93, p. 49] contains the details.

Cross out two columns corresponding to faces that do not share a border and take the determinant. The result is the Alexander polynomial up to a multiple of t^n.

The Alexander Polynomial

Creating the Alexander matrix runs in $O(N^2)$ time, where N is the number of crossings. The way determinants are taught in linear algebra yields a terrifying $O(N!)$ computation, but LU decomposition can speed this up to $O(N^3)$. The Alexander polynomial is hence a polynomial time invariant.

We can speed the computation up even further using tangles ([Bar15]) and implementations exist in [Cul23] and [Mag23c].

The Alexander Polynomial

Having such a quick computation means we can tabulate the invariant for the 352+ million prime knots up to crossing number 19 in less than a day.

This speed comes at a cost, the invariant is not very good at distinguishing knots. Many distinct knots have the same Alexander polynomial. Furthermore, we have the following classical theorem (see [Kaw96]).

Theorem

If $p \in \mathbb{Z}[t, t^{-1}]$ is a Laurent polynomial such that $p(t) = p(t^{-1})$ and $p(1) = \pm 1$, then there is a knot K such that $\Delta_K(t) = p(t)$ where Δ_K is the Alexander polynomial of K.

So Alexander polynomials are not uncommon.

The Jones Polynomial

The Jones polynomial is a knot invariant that is often called *stronger* than the Alexander polynomial, but the two are not directly comparable. There are knots with the same Alexander polynomial but different Jones polynomials (6_1 and 9_{46} in the Rolfsen table), but there are also knots with the same Jones polynomial and different Alexander polynomials (the Figure-Eight 4_1 and the non-alternating knot $K11N19$).

At the end of this talk we will justify why the Jones polynomial is said to be stronger, but keep those examples in your back pocket!

The Jones Polynomial

The original definition uses the Temperley-Lieb algebra, braid groups, and the Markov trace [Jon14], but it is computationally beneficial to think in terms of the Kauffman bracket. This is defined in terms of smoothings of crossings.

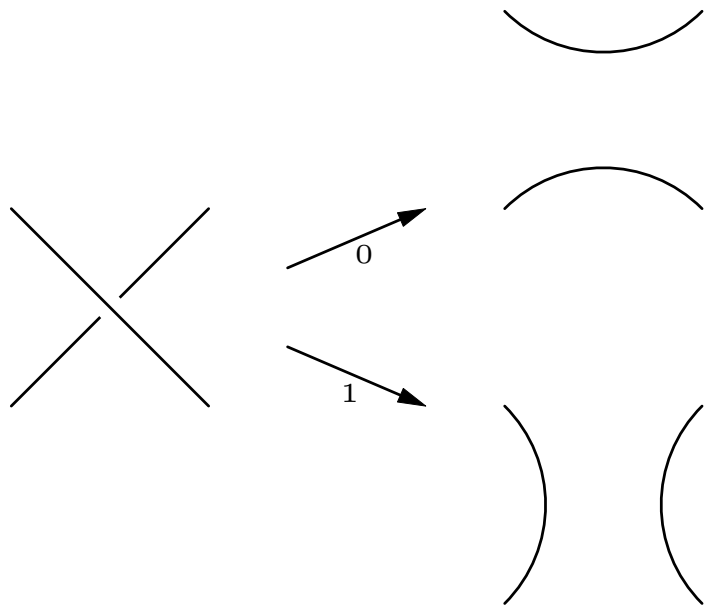

Figure: Smoothing a Crossing

The Jones Polynomial

The Kauffman bracket $\langle L \rangle$ of a link diagram L is defined recursively as follows[1]

$$\langle \emptyset \rangle = 1 \tag{1}$$

$$\langle \mathbb{S}^1 \sqcup L \rangle = (q + q^{-1}) \langle L \rangle \tag{2}$$

$$\langle L \rangle = \langle L_{n,0} \rangle - q \langle L_{n,1} \rangle \tag{3}$$

$$\tag{4}$$

where $\mathbb{S}^1 \sqcup L$ denotes the disjoint union of a circle and L unlinked in $\mathbb{R}^3$, $L_{n,0}$ denotes the zero smoothing at the n^{th} crossing, and similarly $L_{n,1}$ denotes the one smoothing at the n^{th} crossing.

[1] This follows the conventions in [Bar02] which differ slightly from the original in [Kau87].

The Jones Polynomial

This is indeed well-defined and invariant under the type II and type III Reidemeister moves. Type I moves scale the result by q^{-1} and $-q^2$, depending on the sign of the introduced crossing.

There are two ways to make the resulting polynomial invariant under type I moves. We can normalize by the writhe, writing

$$J_L(q) = (-q)^{n_+ - 2n_-} \langle L \rangle \tag{5}$$

where n_+ and n_- are the number of positive and negative crossings in the diagram, respectively.

Alternatively we could scale $\langle L \rangle$ by the monomial q^k such that the zeroth term has degree zero. That is, multiply by $q^{-\mathrm{mindeg}(\langle L \rangle)}$ so that the result is an actual polynomial, instead of a Laurent polynomial.

The Jones Polynomial

There are pros and cons to both definitions. The far more common definition, J_L, has the nice property that $J_L(q) = J_{m(L)}(q^{-1})$ where $m(L)$ is the mirror of L, the result of composing the embedding for L with the reflection $z \mapsto -z$.

The alternative definition means we can skip a writhe computation, but this runs in $O(N)$ so it is not a big deal. However, since $p(q) = q^{-\mathrm{mindeg}(\langle L \rangle)} \langle L \rangle$ is a valid polynomial, this alternative method defines a meromorphic map $z \mapsto z - p(z)/p'(z)$ and induces a Newton fractal in the complex plane with Newton basins and a Julia set.

The topology of these sets are dependent only on the polynomial p, and are hence knot invariants themselves.

The Jones Polynomial

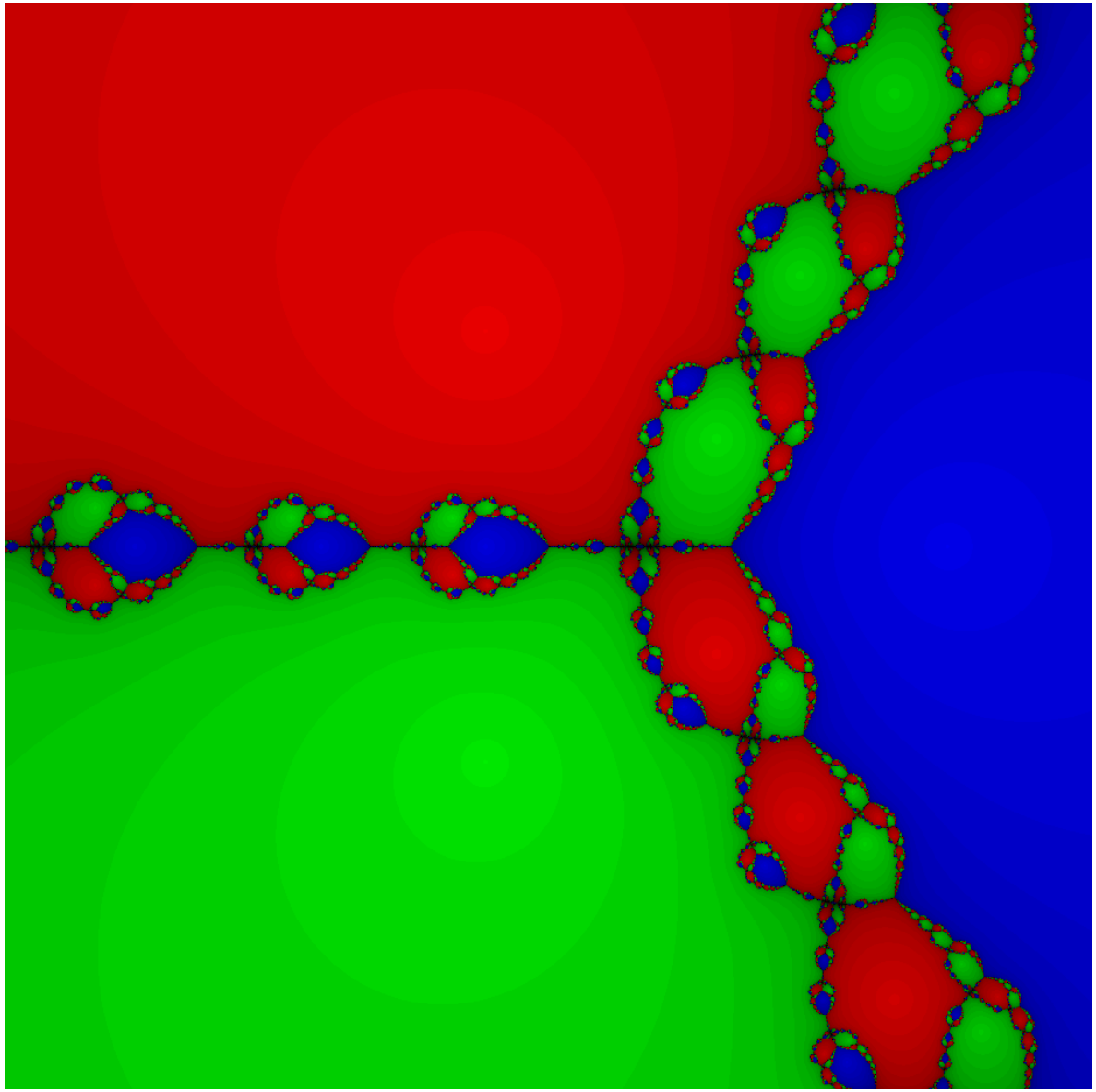

Figure: Newton Fractal for the Right-Handed Trefoil

The Jones Polynomial

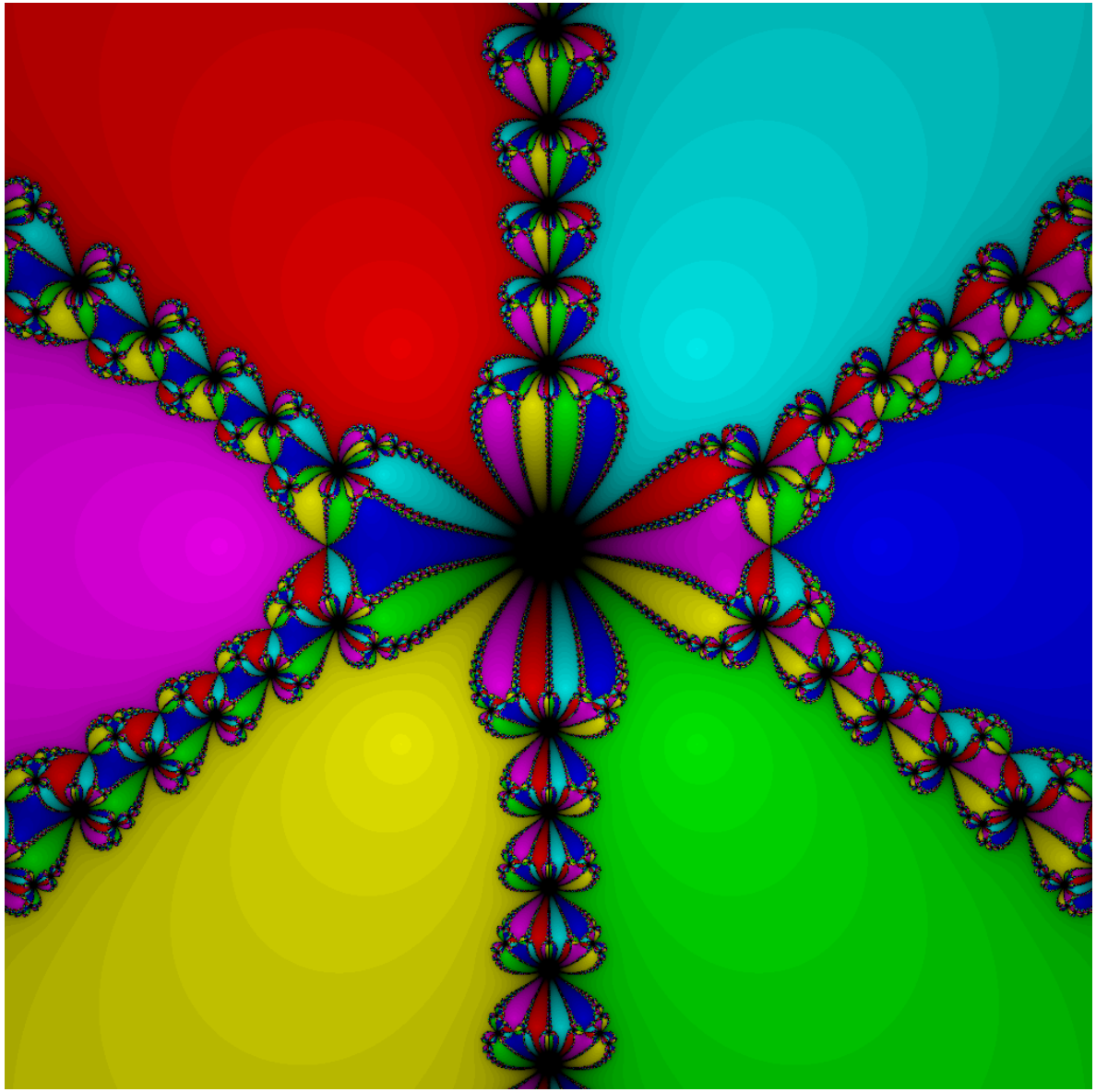

Figure: Unreduced Newton Fractal for the Right-Handed Trefoil

The Jones Polynomial

Computationally the Jones polynomial is at most $O(N)$ away from the Kauffman bracket, so we will restrict our algorithmic efforts to computing $\langle L \rangle$.

The definition gives us a recursive algorithm that is extremely simple to implement. This can be done in just a handful of lines using `C` or `Python`. The simplicity comes at a cost, the algorithm is not only exponential in time, but also in space. After about a dozen crossings one may need several gigabytes of memory for the computation, and 17 crossings and higher is not feasible on most personal computers.

The recursion tree can be expanded into an iterative formula using induction. For your N crossing link diagram each number between 0 and $2^N - 1$ uniquely corresponds to a smoothing of all crossings in the figure.

Given $0 \leq n \leq 2^N - 1$ write n in binary. If the m^{th} bit is a zero, perform a zero smoothing at the m^{th} crossing. Otherwise perform a one smoothing. Do this for each bit in the number.

The Kauffman bracket is then computed via

$$\langle L \rangle = \sum_{n=0}^{2^N-1} (-q)^{w(n)} (q + q^{-1})^{c(n)} \tag{6}$$

where $w(n)$ is the *Hamming weight*, the number of ones in the binary expansion of n, and $c(n)$ is the *cycle counting function*, which counts the number of cycles in the plane that result from the complete smoothing corresponding to the number n.

The Jones Polynomial

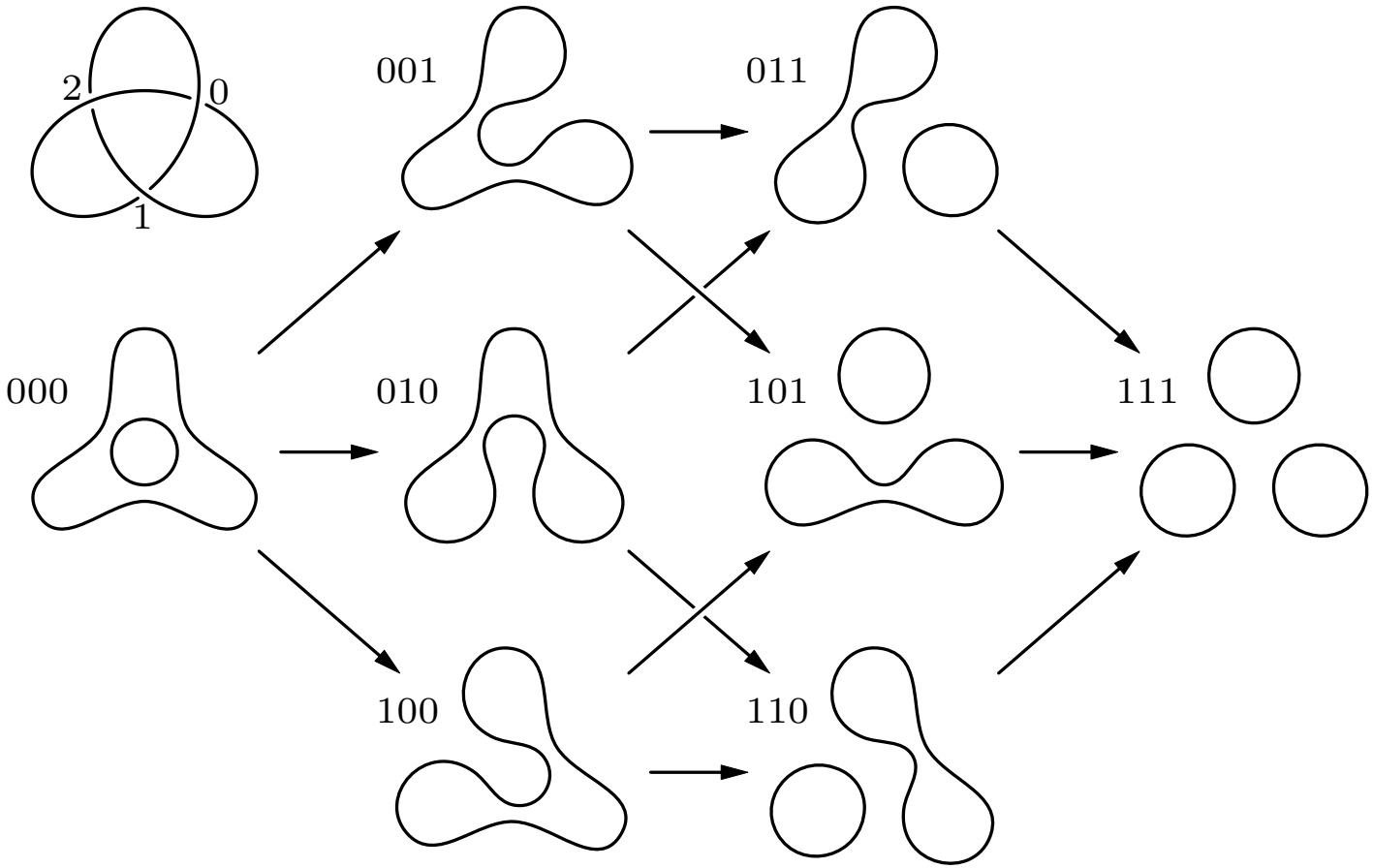

Figure: Cube of Resolutions for the Trefoil

The Jones Polynomial

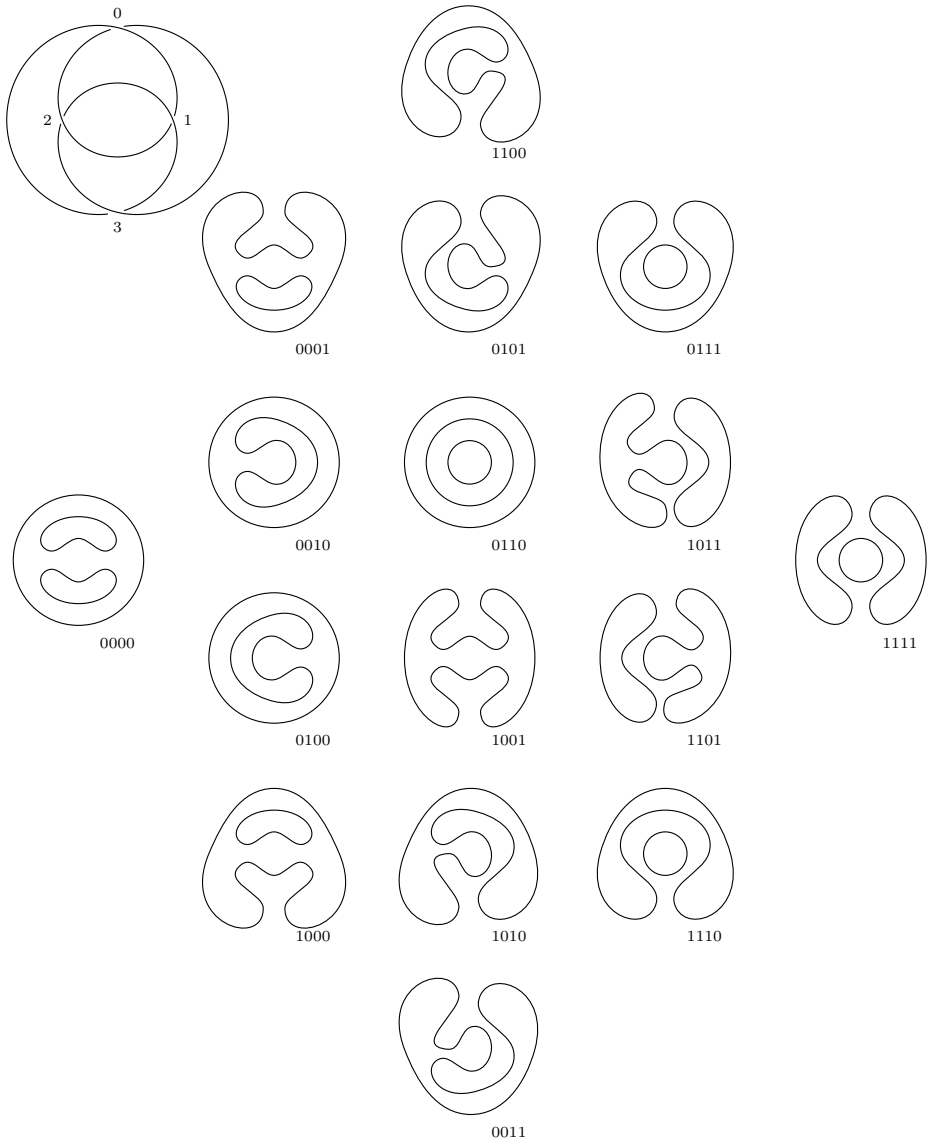

Figure: Cube of Resolutions for the Figure-Eight

The Jones Polynomial

By representing knots and links with extended Gauss codes we can use this summation to devise explicit algorithms for the Kauffman bracket. This is done in [Mag23a].

While still running in $O(2^N)$ time the space requirements have been drastically reduced to $O(N)$. A 32 crossing knot can be dealt with on a laptop, though the computation will take several minutes to a few hours or days, depending on the CPU.

The Jones Polynomial

This invariant has been tabulated for all prime knots up to 19 crossings in [Magb] using the algorithms in [Mag23a] and [Bur18].

We take a moment to outline one more algorithm since it will be of use to us for the computation of Khovanov homology. It uses a *symbolic calculus* by manipulating planar diagram codes, or PD codes. So first, what are PD codes?

The Jones Polynomial

Take a knot diagram and orient it, labeling the arcs from 0 to $2N-1$ in increasing order. Trace your finger along the drawing and at each under crossing write down `X[a,b,c,d]` where `a` is the arc behind you, `b` is to your right, `c` is the arc in front, and `d` is the one on the left.

The result is a string of N 4−tuples. The algorithm starts by observing what the Kauffman relation does to PD code.

The Jones Polynomial

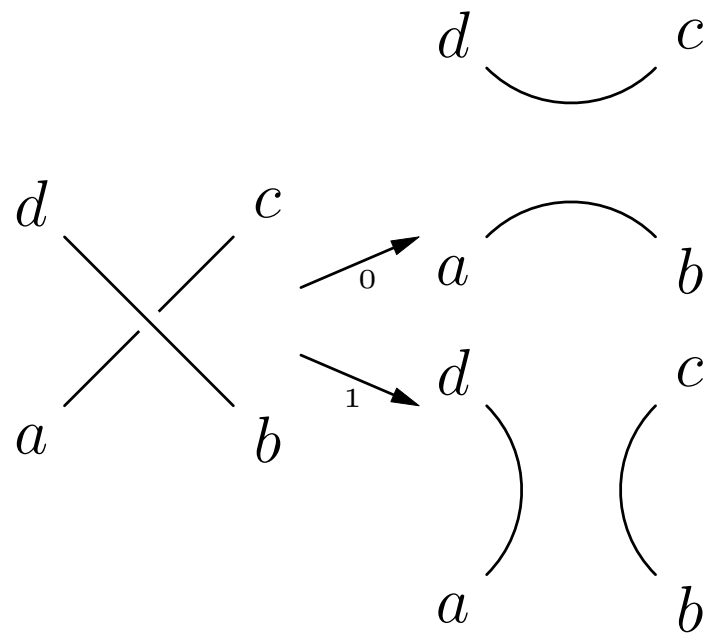

Figure: Resolving a Crossing with PD Code

The Jones Polynomial

The Kauffman relation tells us to perform the symbolic replacement

$$\texttt{X[a,b,c,d]} \mapsto \texttt{P[a,b]P[c,d]} - q\texttt{P[a,d]P[b,c]} \tag{7}$$

We do this for every crossing and obtain the product

$$\langle L \rangle = \prod_{k=n}^{N-1} (\texttt{P[an,bn]P[cn,dn]} - q\texttt{P[an,dn]P[bn,cn]}) \tag{8}$$

where `X[an,bn,cn,dn]` represents the n^{th} crossing.

The Jones Polynomial

After expanding this we obtain a polynomial whose coefficients are formal products of ordered pairs. We reduce further and obtain a Laurent polynomial with integer coefficients by appealing to the Kauffman relations.

Should we see the product `P[a,b]P[a,b]` we may replace it with `P[a,b]` since both expressions tell us the arcs `a` and `b` are now connected. Similarly, we may replace `P[a,b]P[b,c]` with `P[a,c]`.

Lastly, `P[a,a]` tells us we have a cycle. Should we find such an expression we replace it with $q + q^{-1}$.

The Jones Polynomial

Expanding the product on the previous page results in 2^N terms, meaning this algorithm is still exponential. Fortunately there is a trick to significantly speed up our computation.

Instead of expanding the product all at once, we pick a crossing and perform the smoothing $\texttt{X[a,b,c,d]} \mapsto \texttt{P[a,b]P[c,d]} - q\texttt{P[a,d]P[b,c]}$. We then pick the crossing that has the most arcs in common with `X[a,b,c,d]` and smooth that one. That is, we look for the crossing `X[e,f,g,h]` such that `a,b,c,d` and `e,f,g,h` have as many terms in common as possible.

The Jones Polynomial

We combine the products and simplify using the previous relations. By adding the crossings in such a way we grow our *computational front* as slowly as possible, meaning we have less work to do in the end.

This idea is implemented in [Kata] and [Mag23a] and gives a considerable speed boost for many knot diagrams.

Khovanov Homology

The other benefit is that it can be generalized and allow us to compute Khovanov homology. Khovanov homology is a link invariant that associates a chain complex to a link diagram, the homology of which is a link invariant.

Following [Bar02] we again consider the cube of resolutions and associate to each vertex a vector space (or module) $V^{\otimes c(n)}$ where V is a graded vector space (or free module) of graded dimension $q + q^{-1}$. The chain groups are obtained by direct sums over the vertices of the same Hamming weight. The chain maps are defined by summing over local maps defined on the edges of the cube.

The Jones Polynomial

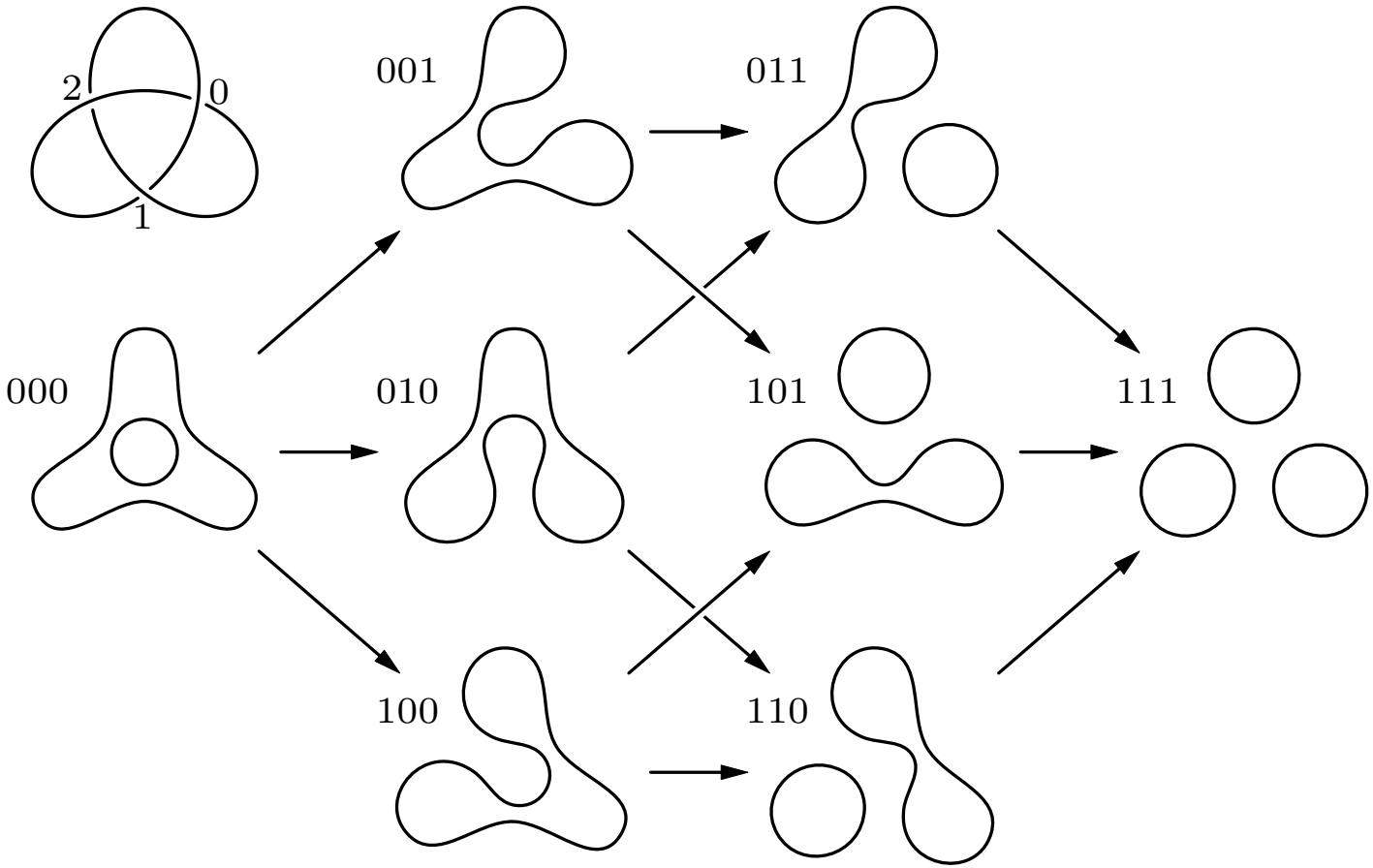

Figure: Cube of Resolutions for the Trefoil

Khovanov Homology

The resulting homology can be decomposed into homogeneous parts [Katb], and the *Khovanov polynomial* is defined by the resulting Poincaré polynomial

$$Kh_L(q,\, t) = \sum_{n,\, m} q^n t^m \dim\big(Kh_m^n(L)\big) \tag{9}$$

The Jones polynomial is recovered via

$$J_L(q) = Kh_L(q,\, -1) \tag{10}$$

In [Bar06] an explicit algorithm is outlined for the computation and this is implemented in [Gre23].[2]

[2] Work in progress implementations in `C` can be found in [Mag23b] and [Mag23c].

Khovanov Homology

The Khovanov polynomial of all prime knots up to 17 crossings has been tabulated and is available in [Magc].

The computation took 6 weeks using fairly powerful hardware. 19 crossings would have taken about 3 years. Optimizations for low crossing knots are underway, as is experimenting with parallelization. This may allow the computation to be done in months, instead of years.

HOMFLY-PT Polynomial

The final invariant to be experimented with is the HOMFLY-PT polynomial. Like Khovanov homology, it generalizes the Jones polynomial, but also generalizes the Alexander polynomial. It is defined via

$$P(\mathbb{S}^1) = 1 \tag{11}$$

$$\alpha P(L_{n,+}) - \alpha^{-1} P(L_{n,-}) = zP(L_{n,0}) \tag{12}$$

Where $L_{n,+}$ denotes the n^{th} crossing as a positive crossing, $L_{n,-}$ denotes swapping the strands over each other to a negative crossing, and $L_{n,0}$ is the orientation preserving smoothing.

HOMFLY-PT Polynomial

The Jones and Alexander polynomial, J_K and Δ_K, respectively, are recovered via

$$J_K(q) = P(q^{-2}, q - q^{-1}) \tag{13}$$

$$\Delta_K(q) = P(1, q - q^{-1}) \tag{14}$$

So the computation is at least as hard as the Jones polynomial, and indeed it is **NP-Hard** [Jae88].

HOMFLY-PT Polynomial

Three algorithms exist, of which I'm still in the early stages of experimenting.

1. Kauffman's skein template algorithm [Kau90].
2. Gouesbet, Meunier-Guttin-Cluzel, and Letellier's algorithm using Gauss codes [Gou99].
3. Burton's modification of Kauffman's algorithm [Bur18].

Burton's algorithm gives the best performance, and he provided a C++ implementation as well [Bur24]. Slight experiments with this code have led to the tabulation of the HOMFLY-PT polynomial of all prime knots up to 19 crossings [Maga].

Results

Keyword	Description
Cr	Crossing number, largest number of crossings considered.
Unique	Number of polynomials that occur for one knot.
Almost	Number of polynomials that occur for two knots.
Total	Total number of distinct polynomials in list.
Knots	Total number of knots in list.
FracU	Unique / Total
FracT	Total / Knots
FracK	Unique / Knots

Table: Legend for the Statistics Table

Results

Cr	Unique	Almost	Total	Knots	FracU	FracT	FracK
03	1	0	1	1	1.000000	1.000000	1.000000
04	2	0	2	2	1.000000	1.000000	1.000000
05	4	0	4	4	1.000000	1.000000	1.000000
06	7	0	7	7	1.000000	1.000000	1.000000
07	14	0	14	14	1.000000	1.000000	1.000000
08	35	0	35	35	1.000000	1.000000	1.000000
09	84	0	84	84	1.000000	1.000000	1.000000
10	223	13	236	249	0.944915	0.947791	0.895582
11	626	77	710	801	0.881690	0.886392	0.781523
12	1981	345	2420	2977	0.818595	0.812899	0.665435
13	6855	1695	9287	12965	0.738129	0.716313	0.528731
14	25271	7439	37578	59937	0.672495	0.626958	0.421626
15	105246	35371	170363	313230	0.617775	0.543891	0.336002
16	487774	173677	829284	1701935	0.588187	0.487260	0.286600
17	2498968	894450	4342890	9755328	0.575416	0.445181	0.256164
18	13817237	4863074	24116048	58021794	0.572948	0.415638	0.238139
19	82712788	27409120	141439472	352152252	0.584793	0.401643	0.234878

Table: Statistics for the Jones Polynomial

Results

This raises the question, if K_n is the number of prime knots with at most n crossings, and if J_n is the number of unique Jones polynomials for prime knots up to n crossings, does J_n/K_n converge to zero?

Results

Cr	Unique	Almost	Total	Knots	FracU	FracT	FracK
03	1	0	1	1	1.000000	1.000000	1.000000
04	2	0	2	2	1.000000	1.000000	1.000000
05	4	0	4	4	1.000000	1.000000	1.000000
06	7	0	7	7	1.000000	1.000000	1.000000
07	14	0	14	14	1.000000	1.000000	1.000000
08	35	0	35	35	1.000000	1.000000	1.000000
09	84	0	84	84	1.000000	1.000000	1.000000
10	225	12	237	249	0.949367	0.951807	0.903614
11	641	71	718	801	0.892758	0.896380	0.800250
12	2051	326	2462	2977	0.833063	0.827007	0.688949
13	7223	1636	9539	12965	0.757207	0.735750	0.557115
14	27317	7441	39222	59937	0.696471	0.654387	0.455762
15	118534	36867	182598	313230	0.649153	0.582952	0.378425
16	578928	187639	919835	1701935	0.629382	0.540464	0.340159
17	3167028	1001101	5033403	9755328	0.629202	0.515965	0.324646

Table: Statistics for the Khovanov Polynomial

Results

We can ask the same question for Khovanov homology. Labeling Kh_n and K_n in a similar way, does Kh_n/K_n converge to zero?

Note, as expected, $Kh_n > J_n$. This simply reiterates the fact that $Kh_L(q, -1) = J_L(q)$ for any link L.

While 4_1 and $K11n19$ have the same Jones polynomial (but different Alexander polynomials), their Khovanov polynomials differ, further cementing the claim that the knots are different.

Results

Cr	Unique	Almost	Total	Knots	FracU	FracT	FracK
03	1	0	1	1	1.000000	1.000000	1.000000
04	2	0	2	2	1.000000	1.000000	1.000000
05	4	0	4	4	1.000000	1.000000	1.000000
06	7	0	7	7	1.000000	1.000000	1.000000
07	14	0	14	14	1.000000	1.000000	1.000000
08	35	0	35	35	1.000000	1.000000	1.000000
09	84	0	84	84	1.000000	1.000000	1.000000
10	241	4	245	249	0.983673	0.983936	0.967871
11	730	34	765	801	0.954248	0.955056	0.911361
12	2494	210	2724	2977	0.915565	0.915015	0.837756
13	9475	1302	11044	12965	0.857932	0.851832	0.730814
14	39401	7170	48329	59937	0.815266	0.806330	0.657374
15	186799	38833	238614	313230	0.782850	0.761785	0.596364
16	979987	209669	1266261	1701935	0.773922	0.744013	0.575808
17	5559808	1157938	7175287	9755328	0.774855	0.735525	0.569925
18	33722920	6480965	42857755	58021794	0.786857	0.738649	0.581211
19	213355372	36387952	264839694	352152252	0.805602	0.752060	0.605861

Table: Statistics for the HOMFLY-PT Polynomial

Results

Most surprising, defining H_n to be the number of unique HOMFLY-PT polynomials for prime knots up to n crossings, we see that H_n/K_n is not monotonic. Also surprising is how high the ratio is for 19 crossings.

It would be intriguing to know what the $\lim\sup$ and $\lim\inf$ are.

Results

Future work

1. Finish FastKH reimplementation.
   - Parallelize batch computations.
   - Parallelize individual computations?
   - Avoid arbitrary precision arithmetic.
   - `C` vs. `Java`, will there be noticeable performance differences?
2. Rewrite Alexander algorithms.
   - The ones used are in `Sage`, `Python`, and `Mathematica`. My `C` routines are in the early stages. Gotten nice speed improvements so far.
3. Fixed parameter tractability for Khovanov homology?
4. Speed tests!
   - More than half a dozen Jones polynomial algorithms.
   - A few Alexander polynomial algorithms.
   - Three HOMFLY-PT algorithms.
   - Which are the fastest for general knots?
5. Do all of this again for 20 crossings!

The End

Thank You!

References I


Bar-Natan, Dror.
On Khovanov's Categorification of the Jones Polynomial.
Algebraic and Geometric Topology, pages 337–370, 2002.


Bar-Natan, Dror.
Fast Khovanov Homology Computations.
Journal of Knot Theory and Its Ramifications, 16:243–255, 2006.


Bar-Natan, Dror.
Polynomial Time Knot Polynomials.
`https://www.math.utoronto.ca/drorbn/Talks/Aarhus-1507/`, 2015.

References II

Burton, Benjamin A.
The HOMFLY-PT Polynomial is Fixed-Parameter Tractable.
In Bettina Speckmann and Csaba D. Tóth, editor, *34th International Symposium on Computational Geometry (SoCG 2018)*, volume 99 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Burton, Benjamin A. and Budney, Ryan and Pettersson, William and others.
Regina: Software for Low-Dimensional Topology.
`http://regina-normal.github.io/`, 1999–2024.

References III


Culler, Marc and Dunfield, Nathan M. and Goerner, Matthias and Weeks, Jeffrey R.
SnapPy, a Computer Program for Studying the Geometry and Topology of 3-Manifolds.
Available at `http://snappy.computop.org`, 2023.


Gouesbet, G. and Meunier-Guttin-Cluzel, S. and Letellier, C.
Computer evaluation of homfly polynomials by using gauss codes, with a skein-template algorithm.
Applied Mathematics and Computation, 105(2):271–289, 1999.


Greene, Jeremy and Bar-Natan, Dror and Pultsin, Nikolay.
JavaKh.
`https://github.com/geometer/JavaKh-v2`, 2023.

References IV


Jaeger, Francois.
Tutte Polynomials and Link Polynomials.
Proceedings of the American Mathematical Society, pages 647–654, 1988.


Jones, Vaughan.
The Jones Polynomial for Dummies.
2014.


Knot Atlas Jones Polynomial.
`http://katlas.org/wiki/The_Jones_Polynomial`.
Accessed: 2024-02-01.


Knot Atlas Khovanov Homology.
`http://katlas.org/wiki/Khovanov_Homology`.
Accessed: 2022-03-10.

References V


Kauffman, Louis.
State Models and the Jones Polynomial.
Topology, 26(3):395–407, 1987.


Kauffman, Louis.
State Models for Link Polynomials.
L'Enseignement Mathématique, 36(2):1–37, 1990.


Akio Kawauchi.
A survey of knot theory.
Birkhäuser Verlag, 1996.


Lickorish, W.
An Introduction to Knot Theory.
Springer, 1997.

References VI


Livingston, Charles.
Knot Theory.
Mathematical Association of America, 1993.


Maguire, Ryan.
HOMFLY-PT Polynomial Database.
http://knots.dartmouth.edu/homfly_polynomial/.
Accessed: 2023-10-16.


Maguire, Ryan.
Jones Polynomial Database.
http://knots.dartmouth.edu/jones_polynomial/.
Accessed: 2023-10-10.


Maguire, Ryan.
Khovanov Polynomial Database.
http://knots.dartmouth.edu/khovanov_polynomial/.
Accessed: 2023-10-10.

References VII


Maguire, Ryan.
Jones Polynomial Implementations.
https://github.com/ryanmaguire/jones_polynomial/, 2023.


Maguire, Ryan.
Knot Data.
https://github.com/ryanmaguire/knot_data/, 2023.


Maguire, Ryan.
The Mathematicians Programming Library.
https://github.com/ryanmaguire/libtmpl/, 2023.