THE 250 KNOTS WITH UP TO 10 CROSSINGS

D. BAR-NATAN

A. KHESIN

(Received TODO DATE)

The list of knots with up to 10 crossings is commonly referred to as the Rolfsen Table. The knots in the table are all unique up to composition, reflection, and the three Reidemeister moves. The approach used to reconstruct this list was very similar to that used by TODO AUTHORS in TODO SOURCE. It involved generating all planar diagrams with up to 10 crossings, and applying layer after layer of simplifications by constructing equivalences between groups of diagrams while also using each diagram with n crossings to generate 2^{n-1} knots with all possible crossings orientations.

I INTRODUCTION

In TODO SOURCE TODO AUTHORS introduce notation to represent a knot with n crossings with n numbers. This notation will be referred to as a DT code. The three Reidemeister moves will be referred to as R1, R2, and R3, respectively. If one were to travel along a knot with n crossings, labeling each strand with a number from 1 to $2n$ each time they came across a crossing, they would find that they have grouped these numbers into n pairs. Each pair would contain the indices of the two strands in a given crossing. It can be observed that each pair will contain one even number and one odd number. Thus, ordering the pairs by their odd number will construct a list of even numbers which can uniquely reconstruct the pairings.

For example, no matter how one chooses a starting point and direction on the trefoil, they will find that 1 is paired with 4, 2 with 5, and 3 with 6. Thus, the pairs can be ordered by their odd number to get (1, 4), (3, 6), and (5, 2). Thus, this set of pairs can be reconstructed with the sequence (4, 6, 2). Since this sequence contains only even numbers, storing half of each value is sufficient and makes computation easier. Thus, the trefoil is represented by (2, 3, 1).

At this point, several things should be noted. TODO AUTHORS did not take half of the resulting values.. This was done here for ease of computation. Additionally, it may be noted that as described so far, the sequence has no way of restoring the orientation of each crossing. Thus, the negative of the corresponding value in the sequence is written if in its crossing, the even strand was above. When any knot is flipped, all the numbers in its sequence become negated, so the first element of the sequence is always written with a positive sign. As a result, if every knot with n crossings can be represented by a signed permutation of the numbers from 1 to n.

II ALTERNATING KNOTS

If a sequence is made up entirely of positive entries, the resulting knot will be alternating. Thus, it is sufficient to generate a reduced list of alternating knots before constructing the non-alternating knots by flipping the crossings.

There are clearly fewer alternating knots than there are positive permutations. Therefore, some of the permutations must be eliminated. A permutation is eliminated if it does not meet any of the following four conditions:

- The permutation must be lexicographically minimal over starting points and directions of enumeration. It is clear that the same knot can produce different permutations depending on where we start numbering and in which direction we proceed. There are $4n$ ways to choose both a starting point and direction (though for cases like (2, 3, 1), all such choices result in the same permutation). If a permutation satisfies this condition it is called minimal.
- The resulting knot must be prime, meaning it cannot be a composition of two knots. For permutations, this means that there must not exist a sequence of $2k$ consecutive numbers mod $2n$ where each number is paired with a different number in that sequence. This also handily eliminates knots that contain a kink and can be simplified with R1 (R3 and the simplifying R2 cannot occur in alternating knots).
- The permutation must be lexicographically minimal over all minimal permuta-

tions of knots connected to it via flypes (see Figure TODO FIGURE).

- The permutation must encode a diagram which is realizable. This means that there must be to draw the knot without adding any intersections beyond the ones encoded in the permutation. The simplest permutation that fails this test is (2, 4, 1, 5, 3).

The first two conditions can be used to avoid checking all such permutations. If one were to arrange all the permutations lexicographically, then went along checking each one, it would be possible to often jump $k!$ permutations ahead instead of just one. This would be because if there is a digit whose difference from its pair is smaller than the difference between 1 and the first digit, then all permutations with that digit in that position will not be minimal. Similarly, if a knot is not prime, this will be represented by a few consecutive digits in the permutation, meaning all permutations obtained by only rearranging the last few digits can be skipped.

The third condition can be checked by using a graph searching algorithm such as breadth first search to find all knots that are connected to a given knot via flypes. All knots except the one that is minimal lexicographically are eliminated from the list of candidates. A flype is embedded in a DT code when there are two sequences of consecutive numbers that, like earlier, are only paired with other elements of those two strings, such that there is a crossing that contains one number from one endpoint of each sequence. This crossing becomes the crossing that gets moved to the other side of the sequences using the flype.

Finally, the fourth condition can be checked using a planarity algorithm for graphs. However, if a 4-valent graph were to be constructed out of a knot by replacing each crossing with a vertex and each strand with an edge, then typical planarity tests would frequently give false positives. That is to say, if there are four strands surrounding a crossing, ignoring rotational symmetry, there are only 2 ways of arranging them for a knot, but there are 6 ways of doing so for an actual graph. The example from earlier involving the permutation (2, 4, 1, 5, 3) is one such case. This permutation does not form a planar knot, but the graph that is created by making the appropriate connections is, strictly speaking, planar.

The reason for cases such as these is due to the fact that a graph might only be planar when the edges around a particular vertex are aligned such that if one were to travel along the knot, they would exit a vertex along an edge adjacent to the one they used to enter it. This would not be allowed in a knot as the whole point of a crossing is that one ends up opposite their point of entry.

To solve this problem, it is sufficient to replace each vertex with four vertices in a diamond pattern (see Figure TODO FIGURE). This would preserve the planarity of the two allowable configurations but would bar the other four, as the diamond would be transformed into a non-planar bowtie shape.

Having eliminated all the knots that do not satisfy the four conditions, a complete list of all of the alternating knots has been constructed.

III Non-Alternating Knots

After generating all the alternating knots, each (except the first) crossing of the knot is toggled to generate $2^{n-1}-1$ non-alternating knots.

Euler's formula generates a very nice relation between the number of faces, edges, and vertices in a connected graph.

$$F - E + V = 2 \qquad [1]$$

Where:

F = number of faces in graph

E = number of edges in graph

V = number of vertices in graph

If the knot in question has n crossings then $V = n$ and $E = 2n$ so $F = n+2$. One of these faces is the external face, leaving $n+1$ faces that are surrounded by edges. When n is no greater than 10, it is impossible to arrange 11 faces such that to get from the external face to some internal one, one must cross at least 3 edges. In other words, any two vertices in the dual graph of the knot are no more than 2 apart.

This means that any non-alternating knot that has a strand passes over three consecutive crossings or one that passes under three consecutive crossings, then it definitely can be reduced and is eliminated from the list of candidates of knots with n crossings. Additionally, any knots on which a simplifying R2 move

can be performed are also removed. Out of the remaining knots, all of those that can be simplified contain the same structure. If one can perform a 2, 1-pass (see Figure TODO FIGURE) on a knot, then it can be simplified. All such knots are also removed from the list. At this point, the only knots that are left are all knots with n crossings, the only issue is that many are the same.

Here it would be wise to extend the definition of lexicographic orderings to signed permutations. Firstly, a positive permutation always comes before a signed permutation. If two signed permutations were to be compared, the one that would come first would be the one whose absolute value is lexicographically smaller. If the absolute values of the permutations are equal, then the one with the smaller index of its first negative value comes first.

At this point, R3 can be applied to each knot in every possible way. If two knots are found to be equivalent, the one that is lexicographically smaller is kept and the others are discarded. Applying R3 is preferable to other Reidemeister moves as there are far more ways to generate knots by adding crossings with R1 or R2. Another move that preserves crossing number is the 2-pass (see Figure TODO FIGURE). Combined with R3, these two moves can be used to find almost all of the knots that a given knot is equivalent to. There are 250 knots with 10 crossings or fewer and just these two moves reduce the candidate list to 253.

At this point, there are several options to narrow the list down to 250. The first is to introduce more crossing number-preserving moves. The second is to construct the graph of connections into higher crossing numbers. By allowing R1 as a valid move and allowing the crossing number to increase by a maximum of 1, the list is narrowed to 251.

Finally, to split the final pair, it is necessary to allow R2 and for the crossing number to increase by as much as 2. This finds that there was a final pair of equivalent knots and produces the full reduced list of the 250 prime knots with 10 crossings or fewer.

IV Invariants

Since computation time is massively increased by examining knots with more crossings that are connected to a given knot. To avoid searching through the connections of each of the 253 knots, invariants can be used. Thus, the Jones polynomial can be used to check if a given knot's polynomial is unique in the list of candidates. If it is not, the connections of that knot are investigated and any duplicates are excised, eventually whittling the list down to just the knots that have a unique Jones polynomial among the candidates or those whose connection tree has been examined and no other members of the candidate list were found to be connected to it.

To find the Jones polynomial of a given knot, it must be written in planar diagram notation as opposed to in DT code form. To find this notation, it is sufficient to determine for each crossing which of the two possible arrangements of strands around a crossing will result in the whole knot being planar. From earlier, it is clear that a solution exists. To find this solution, it is sufficient to construct a spanning tree of the knot. Then, a linear system of equations can be formed with n variables where each can be one of two choices, one for each of the two edge configurations.

To construct each equation, each pair of edge that was not included in the tree is examined. A closed loop can be formed for each edge in each pair by connecting the vertices the edge was meant to connect by a path inside the tree and by an edge outside of it. Since the loops cannot intersect outside the tree, they must intersect an even number of times inside it. Since the number of intersections is dependent on the configuration at some vertices, a set of linear equations can be formed and then solved to convert the knot into planar diagram notation.