Coloring the Mu Transpososome

I. K. Darcy*[1], J. Chang[3], N. Druivenga[3,4], C. McKinney[2], R. K. Medikonduri[1], S. Mills[3], J. Navarra-Madsen[5], A. Ponnusamy[6], J. Sweet[3,4], T. Thompson[2]

[1]Mathematics Department, University of Iowa, Iowa City, Iowa 52242, USA
[2]undergraduate researcher, University of Texas at Dallas, Summer 2003
[3]undergraduate researcher, University of Texas at Austin, Summer 2004
[4]undergraduate researcher, University of Iowa, 2004-2005
[5]Mathematics Department, Texas Woman's University, Denton, Texas 76204, USA
[6]Credit Suisse First Boston, USA

Email: Isabel K. Darcy *- idarcy@math.uiowa.edu;

*Corresponding author

Abstract

Background: Tangle analysis has been successfully applied to study proteins which bind two segments of DNA and can knot and link circular DNA. We show how tangle analysis can be extended to model any stable protein-DNA complex.

Results: We have developed a computational algorithm to find the topological conformation of DNA bound within a protein complex. The algorithm uses an elementary invariant from knot theory called colorability to encode and search for possible DNA conformations. We apply this algorithm to analyze the experimental results of Pathania, Jayaram, and Harshey (Cell 2002). We show that the only possible DNA conformation bound by Mu transposase is the five crossing solution found by Pathania et al.

Conclusions: Our algorithm combined with the experimental technique described in Pathania et al can be applied to determine the topological conformation of DNA bound within any stable protein-DNA complex.

Background

Tangles have many applications in modeling protein-DNA binding [1–5]. An *n-string tangle* consists of n strings properly embedded in a 3-dimensional ball. Some examples of 2-string tangles and a 3-string tangle are shown in Fig. 1. A protein complex bound to n segments of DNA can be modeled by an n-string tangle. The protein complex is modeled by the 3D ball while the n DNA segments can be thought of as n strings properly embedded in a protein ball (note each string represents one segment of double-stranded DNA). This is an extremely simple model of protein-DNA binding. A ball does not accurately represent the shape of a protein complex, and DNA sometimes winds around a protein complex as opposed to being embedded within the protein complex. However, from this simple model much information can be gained.

When modeling protein-DNA reactions, it is helpful to know how to draw the DNA segments bound by the protein. For example, does the DNA molecule cross itself within the protein complex or does it bend sharply? Tangle analysis can be used to determine the topological shape of the DNA segments bound by a protein complex. Tangle analysis does not determine the exact geometry and hence cannot determine the sharpness of DNA bending, but it can determine the overall topology. This information can be used to infer which DNA sequences are likely to be close to each other [5] as well as other information useful for modeling protein-DNA reactions.

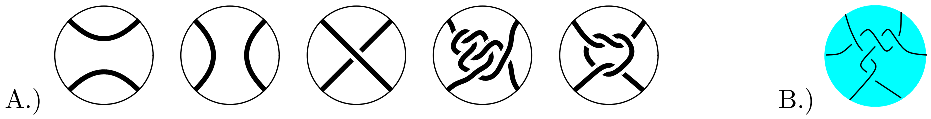


Figure 1: A.) Some 2-string tangles. B.) a 3-string tangle.

The *Mu transpososome* refers to the Mu transposase protein complex (Mu) and the three DNA segments bound by this protein complex. Since three DNA segments are bound by Mu, the Mu transpososome can be modeled by a 3-string tangle. *DNA transposition* is the process where a piece of DNA can change its location within a genome. The Mu transposition pathway involves the formation of a series of protein-DNA complexes: LER, type 0, type 1, type 2 (for more biology background, see [5,6] or (Darcy IK, Luecke J, Vazquez M: Mu Transpososome. Unpublished data)). An experimental technique called *difference topology* [5,7,8] combined with tangle analysis was used in [5] to determine that the LER and type 0 Mu-DNA complexes can be modeled by the five crossing tangle shown in Fig. 1B. There are an infinite number of tangles that mathematically satisfies these experimental results (Darcy IK, Luecke J, Vazquez M: Mu Transpososome. Unpublished data). These other conformations are very complicated and hence biologically unlikely to model the Mu transpososome, but these leave open the possibility that there are other biologically relevant models.

The conformation determined by [5] has five DNA crossings. It is biologically unlikely that Mu binds more than eight DNA crossings. Currently, there are two methods which can be used to determine the DNA conformation bound within the Mu transpososome by assuming only an upper bound on the number of crossings trapped by Mu. By analyzing four pairs of experiments described in [5], the mathematics of three manifold theory was used to rule out certain types of conformations as well as to determine all possible solutions up to eight crossings (Darcy IK, Luecke J, Vazquez M: Mu Transpososome. Unpublished data). These results apply to proteins binding three segments of DNA if the products of the difference topology experiments (as described below) belong to the family of $(2, k)$ torus knots/links, k even or $k = \pm 3$.

Another method to determine the DNA conformation bound by Mu is computational. We describe a computational algorithm that we have implemented to solve the system of tangle equations modeling the experiments in [5]. For this analysis, we only need the results from three pairs of experiments in [5], but could have implemented additional experimental results if it had been needed.

This software can easily be modified to solve any system of n-string tangle equations up to a certain crossing number. Hence, we can solve any system of tangle equations up to a fixed crossing number including those modeling difference topology experiments applied to any protein complex that stably binds any number of segments of DNA.

Difference topology and tangle modeling

We briefly describe some of the difference topology experiments and tangle model from [5]. For a more detailed description, see [5] or (Darcy IK, Luecke J, Vazquez M: Mu Transpososome. Unpublished data). The idea behind the experimental technique of difference topology is to use a protein such as Cre recombinase to trap crossings bound by the protein under study. This is illustrated in Fig. 2. Mu is represented by the cyan colored ball. To show how a difference topology experiment works, we will assume the DNA conformation bound by Mu contains 5 crossings based upon the results of [5]. In this technique, circular DNA is first incubated with the proteins under study (in this case, Mu = cyan ball). These proteins bind DNA, possibly trapping DNA crossings within the protein complex. A second protein whose mechanism is well understood is added to the reaction (in this case, Cre, represented by smaller pink ball). This second protein, Cre, cuts the DNA and changes the circular DNA topology before resealing the breaks, resulting in knotted or linked DNA. The crossings bound by the first protein, Mu, will affect the product topology. In Fig. 2, four of the five crossings bound by Mu are trapped by the action of Cre, resulting in a four crossing link. Hence, one can gain information about the DNA conformation trapped by the first protein, Mu, by determining the knot/link type of the DNA knots/links produced by the second protein, Cre.

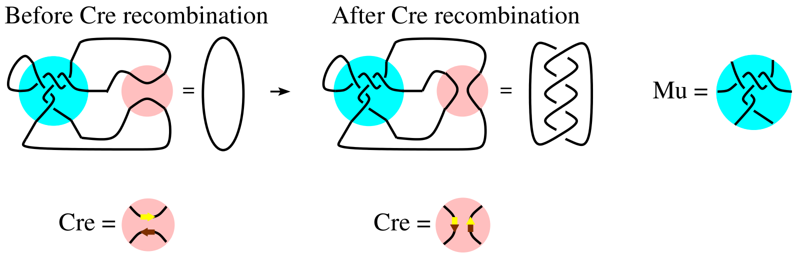


Figure 2: Difference topology experiment. Mu represented by the cyan colored ball is shown bound to five DNA crossings. Cre is represented by the smaller pink ball. Before Cre recombination, the DNA is circular and unknotted. Cre recombination changes the DNA configuration outside of the Mu transpososome. Since four of the five crossings bound by Mu are trapped by Cre recombination, the DNA product configuration equals a four crossing link.

Note that in the substrate configuration, three loops emanate from the Mu transpososome. By choosing which pair of loops to place Cre binding sites, the location of Cre action can be controlled. Six different substrates were constructed in [5] in which Cre binding sites were placed on every pair of loops in two different orientations. Models proposed in [5] of these six reactions are illustrated in Fig. 3. The cyan colored ball still represents the DNA bound by Mu transposase while the pink colored ball still represents the DNA bound by Cre. Crossings within the green annulus represent crossings not trapped by either protein complex. Sometimes an extra crossing not bound by either protein is needed for correct DNA orientation within the Cre protein complex, depending on the orientation of the Cre binding sites on the two loops. When comparing products where the Cre sites are placed on the same pair of loops but in different orientation, it was assumed that the extra crossing occurred with the higher crossing product.

If we do not assume the shape of DNA bound by Mu, the tangle equations corresponding to these experiments is shown in Fig 4 where the tangle T represents the unknown DNA conformation bound by Mu. In two of the experiments, the knot/link product was fully identified. In the remaining four experiments, only the crossing number of the knot/link was determined. There is only one three crossing knot and only one four crossing link up to mirror image. Hence, we know that for the three or four crossing products in Fig. 4, the crossings are either all left-handed or all right-handed.

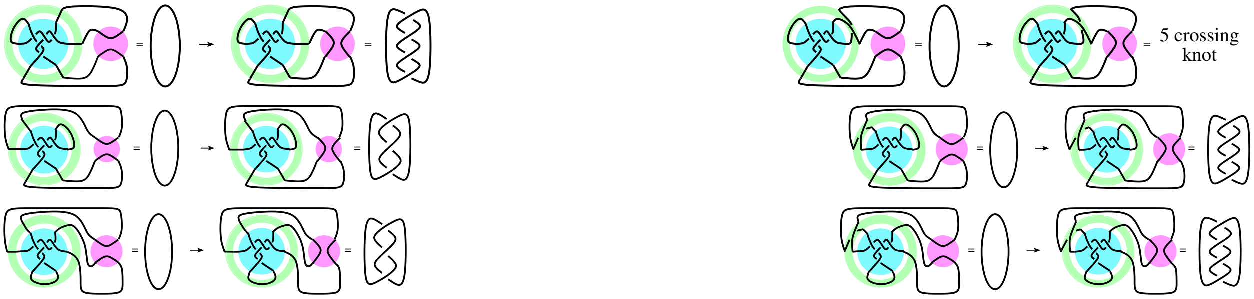


Figure 3: Tangle model from [5].

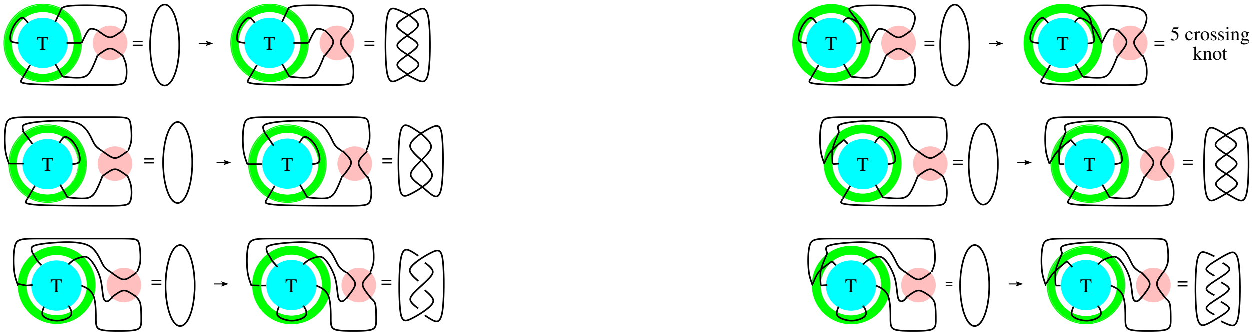


Figure 4: Tangle equations corresponding to difference topology experiments in [5].

Note that the tangle model in Fig. 4 consists of a system of tangle equations with one unknown, the tangle T. The tangle T (cyan ball) represents the unknown DNA conformation bound by Mu. The tangle in Fig. 1B is a solution for T as shown in Fig. 3. We will show that this is the only biologically relevant solution for T.

Mathematical Model: *Determining the DNA conformation bound by Mu is equivalent to solving the system of tangle equations in Fig. 4 for the 3-string tangle T. The solution, however, is a 2-dimensional topological approximation of the 3-dimensional conformation.*

In order to find the Fig. 1B solution, Pathania et al [5] assumed the protein-bound DNA was a 3-branched supercoiled structure like that in Fig. 5. In this figure, the three branches consists of 3, 4, or 5 crossings while the three-branched tangle in Fig. 1B has one branch containing one crossing and two branches containing two crossings. Pathania et al [5] used the number of crossings in the knotted or linked DNA products to determine the number of crossings in each of the three branches. But the question remains if there are any other biologically relevant solutions if we don't assume a 3-branched supercoiled structure.

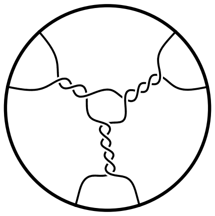


Figure 5: A three-branched tangle.

In the next section, we describe the tangle invariant, colorability, which we use to search for solutions for T. However, a thorough understanding of this invariant is not necessary to understand the main idea behind the algorithm discussed in **Results**.

The coloring invariants

A *diagram*, $D(\mathbf{T})$ of a knot, link, or tangle $\mathbf{T}$ is a projection of $\mathbf{T}$ into $\mathbb{R}^2$ where at a crossing only double points (only two points are superimposed) are allowed and gaps are used to indicate which part of the knot crosses under. Two diagrams correspond to the same 3D knot/link/tangle if one diagram can be converted to the other diagram via a sequence of Reidemeister moves–RI, RII, and RIII (Fig 6).

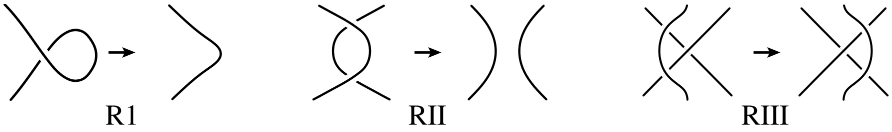


Figure 6: Reidemeister moves.

A *coloring* of a diagram $D(\mathbf{T})$ is a function $C : \{arcs\ of\ D(\mathbf{T})\} \mapsto \mathbb{Z}_m$ where the elements of $\mathbb{Z}_m = \{0, 1, ..., m-1\}$ will be called colors and such that at each crossing the relation $y + z - 2x = 0$ mod m holds, where x is the color assigned to the overarc and y and z are the colors of the two underarcs. See Fig. 7. A coloring is *trivial* if the coloring function is the constant map, i.e., all the arcs are assigned the same value or "color". A knot or link is said to be *m-colorable* if there exists a non-trivial coloring. This is a knot/link invariant in that if one diagram of the knot/link $\mathbf{K}$ is m colorable than all diagrams corresponding to $\mathbf{K}$ are m-colorable [9]. For an elementary introduction to coloring, see [10], (Navarra-Madsen, J. and Darcy, IK: Colorability and n-String Tangles, Unpublished data). We will more thoroughly define how coloring relates to tangles below.

A *coloring matrix* of a knot/link/tangle diagram, $\mathbf{T}$, is any matrix, $\mathbf{M_T}$, which is row equivalent to a

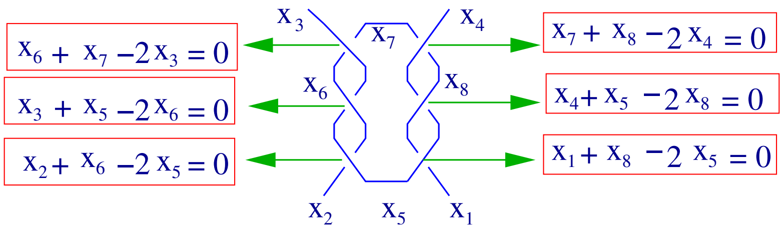


Figure 7: Coloring A 2-string Tangle.

coefficient matrix corresponding to the coloring equations. For example, the 6×8 matrix in equation 1 is a coloring matrix corresponding to the tangle diagram in Fig. 7. Each row corresponds to one of the six crossings in the tangle diagram while each column represents one of the eight arcs, x_5, x_6, x_7, x_8, x_1, x_2, x_3, x_4 in the tangle diagram.

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & -2 & 0 \\ 1 & -2 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & -2 \\ 1 & 0 & 0 & -2 & 0 & 0 & 0 & 1 \\ -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{1}$$

We will call the arcs which have one endpoint on the boundary of the tangle 3-ball *endpoint arcs*. The remaining arcs will be called *interior arcs*. Notice that we place the four columns corresponding to the endpoint arcs, x_1, x_2, x_3, x_4, as the four rightmost columns of the matrix $\mathbf{M_T}$. Take a row echelon form of $\mathbf{M_T}$ using the row operations $r_i \longleftrightarrow r_j$, $r_i \longrightarrow r_i + tr_j$, $i \neq j, t \in \mathbb{Z}$, $r_i \longleftrightarrow -r_i$. Recall that since we are working in Z_m where m is an arbitrary integer, scaling a row is not allowed. An echelon form, $EF(\mathbf{M_T})$ is

$$EF(\mathbf{M_T}) = \left(\begin{array}{cccc|cccc} 1 & 0 & 0 & -2 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 3 & 0 & -1 & -2 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & -3 \end{array}\right) \tag{2}$$

We define the standard echelon form of a matrix, $SF(\mathbf{M})$, to be the echelon form in which each leading entry (first non-zero term in a row) is positive and if a_{ij} is a leading entry of the ith row, then $0 \leq a_{\lambda j} \leq a_{ij} - 1, 1 \leq \lambda < i$. The standard echelon form of a matrix is unique.

If the endpoints' unknowns, x_1, x_2, x_3, x_4 correspond to the four rightmost columns, then $\mathbf{M_l(T)}$ = the lower right hand corner 2×4 submatrix of $\mathbf{M_T}$ in standard echelon form is a tangle invariant. It is a tangle invariant in that if you take two diagrams of the same tangle $\mathbf{T}$ and place the endpoint arcs in the same order in the last columns of their respective coloring matrices, then no matter how the interior arcs are labeled, $\mathbf{M_l(T)}$ will be the same for both diagrams. In addition, the absolute value of the determinant of the upper left 4×4 submatrix, $\mathbf{d_u(T)} = 3$, is also an invariant.

$$\mathbf{M_l(T)} = \begin{pmatrix} 1 & -1 & 1 & -1 \\ 0 & 0 & 3 & -3 \end{pmatrix}, \quad \mathbf{d_u(T)} = 3 \tag{3}$$

For an n-string tangle, $\mathbf{T}$, with a $k \times (k+n)$ coloring matrix $\mathbf{M_T}$ (listing the endpoint arcs in the right-most columns of the matrix), $\mathbf{M_l(T)}$ = the lower right-hand corner $n \times 2n$ submatrix of $\mathbf{M_T}$ in standard echelon form and $\mathbf{d_u(T)}$ = absolute value of the determinant of the upper $(k-n) \times (k-n)$ submatrix of $\mathbf{M_T}$ are both invariants of $\mathbf{T}$.

In order to calculate $\mathbf{M_l(T)}$, we must label the endpoint arcs with distinct variables. If two endpoint arcs correspond to the same arc (i.e., a string does not pass under any other string including itself so that it projects to just one arc), we can doubly label the arc, labeling one endpoint x_i and the other x_j and adding the equation $x_i - x_j = 0$.

Results

We describe a computational algorithm we have implemented to solve the system of tangle equations in Fig. 4. This program can easily be modified to solve any system of n-string tangle equations. Hence we can computationally solve any system of tangle equations up to a fixed crossing number including those modeling difference topology experiments applied to any protein that stably binds any number of segments of DNA.

The algorithm has several steps:

1. Generate tangles up through 8 crossings.

2. Check a topological invariant to determine which of the generated tangles could result in the experimental products.

3. Check if the notation used to encode a tangle actually corresponds to a tangle.

4. Determine if the surviving tangles correspond to equivalent or different tangles.

We first determine how the strings enter and exit the tangle. The parity of a tangle refers to the order in which the strings enter and exit the 3-ball. A solution to the tangle equations in Fig. 4 can have one of two possible parities: the strings enter and exit the tangle as in Fig. 8A or as in Fig. 8B. This is easily determined by noting which of the equations in Fig. 4 involve a one component knot versus a two component link. For example, the string entering in at x_1 cannot exit at x_2 since the top left equation in Fig. 4 involves the one component unknot.

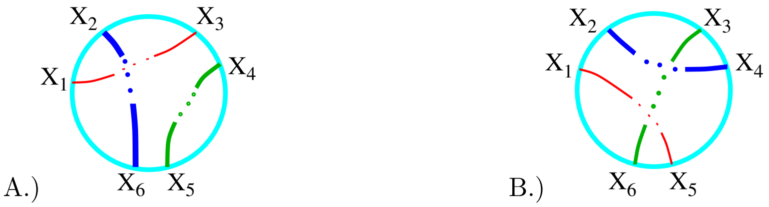


Figure 8: *Possible parities.*

A number of techniques have been used to encode knot diagrams for computational purposes [11, 12]. As described in **Methods**, we use coloring matrices to encode tangle diagrams. We generate matrices which could correspond to tangle diagrams up through eight crossings. We check each matrix to determine if it has the correct coloring invariants to be a solution to the tangle equations in Fig. 4. As shown in table 1, this eliminates the majority of the generated matrices. Not all generated matrices correspond to a tangle. We use an algorithm similar to that described in [13] to remove all matrices which do not correspond to a tangle.

Recall that a tangle can be represented by a number of different diagrams related by Reidemeister moves. Unfortunately, there is no algorithm guaranteed to determine if two tangle diagrams are equivalent. In fact, in order to simplify a diagram, it may be necessary to first increase the number of crossings in the diagram. Thus this software does not determine all tangle equivalences, but does reduce the output sufficiently to handle the remaining possibly equivalent tangles by hand. While generating matrices, we omit matrices where the corresponding diagram can be simplified by RI or RII moves (Fig. 6). A tangle diagram containing the left-hand side of an RIII move will be equivalent to the tangle diagram obtained after the RIII move has been performed. Hence we choose one of these tangle diagrams and discard the other. As discussed in **Methods**, we also perform some other simplifications which involve a combination of RI, RII, and RIII moves. As shown in table 1, this leaves us with 13 matrices: ten with the parity shown in Fig. 8A and three with the parity shown in 8B.

# of Crossings	# of Matrices Generated	Parity Fig 8A			Parity Fig. 8B		
		Col	Draw	Non-Equiv?	Col	Draw	Non-Equiv?
≤ 4	1,639	0	0	0	0	0	0
5	34,578	1	1	1	1	0	0
6	794,578	22	4	0	22	0	0
7	19,781,058	354	15	3	400	0	0
8	537,193,563	5019	106	6	5595	6	3

Table 1: Number of matrices with the correct coloring invariants (Col columns), corresponding to a drawable tangle (Draw columns), and which are potentially non-equivalent (Non-equiv columns). The first column refers to the number of crossings in the tangle diagram. The second column gives the number of matrices generated which could correspond to a tangle with a fixed crossing number. The results in the next three columns assume the parity in Fig. 8A while the results in the last three columns assume the parity in Fig. 8B. The columns labeled "Col" state the number of generated matrices which have the correct coloring invariants to satisfy the equations in Fig. 4. However, not all generated matrices correspond to a tangle. The columns labeled "Draw" give the number of matrices which correspond to a drawable tangle with the correct coloring invariants. The number of these matrices which may correspond to non-equivalent tangles is given in the columns labeled "Non-equiv?". Note, however, that the algorithm does not identify all equivalent tangles.

We checked the remaining thirteen tangles corresponding to these matrices by hand. The ten tangles with Fig. 8A parity are all equivalent to the five crossing tangle found in [5] (Fig. 1B). The three tangles with Fig. 8B parity are all equivalent to one of the two eight crossing tangles in Fig. 9. The eight crossing solutions do not satisfy a fourth pair of experiments from [5] that we have not described in this paper (see [5] or (Darcy IK, Luecke J, Vazquez M: Mu Transpososome. Unpublished data)). Pathania et al would also have determined that these eight crossing solutions are biologically unlikely even without the fourth pair of experiments. Hence there is only one biologically relevant tangle solution up through eight crossings.

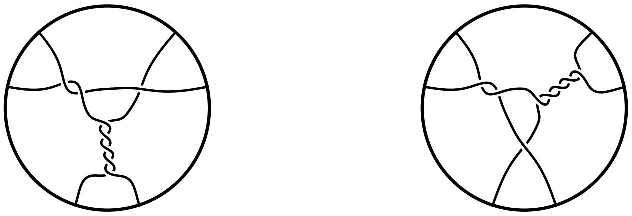


Figure 9: The two possible Fig. 8B parity solutions.

Discussion and Conclusions

Pathania et al [5] needed to assume the basic shape of a 3-branched supercoiled structure (Fig. 5) in order to find the solution shown in Fig. 1b. (Darcy IK, Luecke J, Vazquez M: Mu Transpososome. Unpublished data) proved that this assumption is correct for small crossing tangles when applied to the four pairs of experiments in [5] as well as to similar experiments involving proteins that bind three segments of DNA if products belong to the family of $(2, k)$ torus knots/links, $k = 3$ or k even. The algorithm in this paper needed only the three pairs of experiments illustrated in Fig. 4 to reach the same conclusion as (Darcy IK, Luecke J, Vazquez M: Mu Transpososome. Unpublished data) regarding tangle solutions with eight or fewer crossings. No assumptions regarding the DNA conformation bound by the protein complex are needed except for an upper bound on the number of crossings. This algorithm can also be applied to analyze any difference topology experiment no matter the number of DNA segments bound by the protein complex.

A tangle solution, however, is a 2-dimensional topological approximation of the 3-dimensional structure. It does not determine sharpness of DNA bending, but it does give an important starting point from which other modeling techniques may be applied. No information regarding the Mu-DNA conformation existed before [5]. Since then a partial structure based on scanning transmission electron microscopy (STEM) at cryo-temperatures has become available [14], but this involves only a portion of the protein complex and a change in one of the DNA sequences bound by Mu. Information regarding protein-bound DNA conformations can sometimes be obtained via crystallography, STEM, or FRET (fluorescence resonance energy transfer), but all these techniques are quite difficult and currently can only be applied to small protein-DNA complexes. The experimental technique of difference topology combined with the algorithm described in this paper can be applied to study any stable protein-DNA complex no matter the size of the protein complex or number of DNA segments bound by the protein complex.

Recall that in the Mu tangle model from [5] (Figs 3, 4), it is assumed that at most one crossing is trapped outside of the protein complexes (modeled within the green annulus). Since the Mu and Cre bind to specific DNA sequences, the length of the DNA between the Mu binding sites and Cre binding sites can be controlled. The shortest length needed for the reaction to take place was determined in [5] in order to prevent trapping extraneous crossings. The difference topology experimental technique can also be applied to proteins that bind to arbitrary DNA sequences rather than specific DNA sequences, but the results would not be expected to be as clean. It was shown in [15], that if the length of DNA between binding sites is not properly controlled, then the number of protein-bound DNA crossings may be overestimated. But even if we are left with a 2-dimensional approximation, it is still a significant improvement over having no information on how to draw the DNA in a protein-DNA complex.

We are also not mathematically limited to equations resulting from Cre recombination. Any protein which can change DNA topology could potentially be used in a set of difference topology experiments to obtain a different system of tangle equations. For example topoisomerases change the topology of circular DNA by changing DNA crossings. It may be possible to obtain a more 3-dimensional model by averaging 2-dimensional solutions from two or more systems of tangle equations. Cre, however, may be the easiest to work with due to its sequence specificity and simple mechanism.

To solve a different system of tangle equations, we may only need to change a few lines of code. However, it may also be necessary to add additional tangle invariants and/or equivalence moves. Although coloring is not that powerful of a knot invariant, it is a powerful tangle invariant. It is the only invariant we need to check to determine if a tangle up through eight crossings is a solution to the equations in Fig. 4. However, there is no guarantee that this invariant will be sufficient for a different system of tangle equations. Fortunately, there are a number of other invariants as well as software available for calculating these invariants which can be used if necessary [12, 16].

Our algorithm left us with only 13 different coloring matrices which could correspond to tangle solutions to the system of equations in Fig. 4. We could have added additional equivalence moves to further reduce this output, but it was quicker to check these 13 matrices by hand. For a different system of equations, additional equivalence moves may be needed to reduce the output to a handful of matrices. Additional equivalence moves will be added as needed.

Currently this algorithm takes about two days to find solutions through eight crossings. The number of tangles grows exponentially with crossing number. However, the efficiency of the algorithm can be significantly improved. In particular, this algorithm is easily parallelizable. We will extend this program to solve arbitrary n-string tangle equations up to at least ten crossings. A long-term goal is to create software accessible to those without a background in knot theory. But in the meantime, we can easily modify this algorithm to solve any specified system of tangle equations; hence an experimentalist need not wait for the final version of this software before performing difference topology experiments.

Methods

Tangle generation

We use the coloring matrix of a tangle to encode its shape. Recall that a solution to the tangle equations in Fig. 4 can have one of two possible parities: the strings enter and exit the tangle as in Fig. 10A or as in

Fig. 10B. For tangle generation, we are not placing the endpoint arcs in the rightmost columns. This simplifies the matrix generation as well as determining if a matrix corresponds to a drawable tangle or if two matrices correspond to the same tangle. In order to calculate the coloring invariants, we will later move the columns corresponding to the endpoint arcs to the rightmost columns. The red string which enters in at the point labeled x_1 and exits at the point x_i will be called string 1. The green string which enters in at the point labeled x_{i+1} and exits at the point x_j will be called string 2 while the remaining blue string will be called string 3.

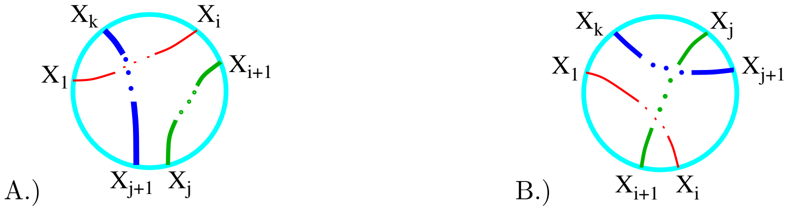
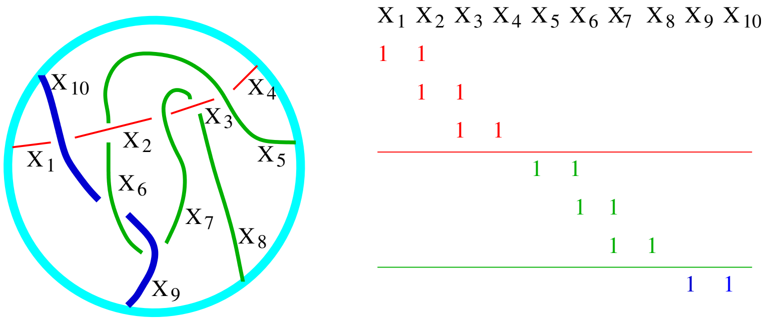


Figure 10: *Possible parities.*

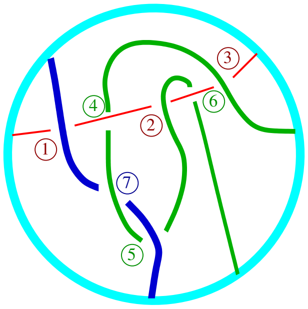
We first consecutively label the arcs of red string 1 beginning with x_1 as illustrated in Fig. 11. The red string is broken into four arcs with the arcs consecutively labeled x_1, x_2, x_3, x_4. We then label the arcs of the green second string starting from the first endpoint arc clockwise from the red endpoint arc x_4. String 2 is broken into four arcs which are consecutively labeled x_5, x_6, x_7, x_8. We then label the arcs of string 3, x_9, x_{10}, starting from the first endpoint arc clockwise from the last labeled arc of string 2.



$$\begin{array}{cccccccccc}
X_1 & X_2 & X_3 & X_4 & X_5 & X_6 & X_7 & X_8 & X_9 & X_{10} \\
1 & 1 & & & & & & & & \\
 & 1 & 1 & & & & & & & \\
 & & 1 & 1 & & & & & & \\
\hline
 & & & & 1 & 1 & & & & \\
 & & & & & 1 & 1 & & & \\
 & & & & & & 1 & 1 & & \\
\hline
 & & & & & & & & 1 & 1
\end{array}$$

Figure 11: *Example: labeling arcs.*

We next label the crossings. Beginning with string 1, we consecutively number the under-crossings (Fig 12). Hence for string 1, crossing number i occurs between string 1 arcs x_i and x_{i+1}. For string 2, crossing number j occurs between string 2 arcs x_{j+1} and x_{j+2} while for string 3, crossing number k occurs between string 3 arcs x_{k+2} and x_{k+3}. This determines the placement of the two '1"s in each row (Figs. 11, 12). To generate matrices that could correspond to a coloring matrix, we can now place one -2 in each row in all possible combinations.



$$\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 \\
0 & 1 & 1 & 0 & 0 & 0 & -2 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & -2 & 0 & 0 & 0 & 0 & 0 \\
0 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & -2 & 0 \\
0 & 0 & -2 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 1 & 1
\end{pmatrix}$$

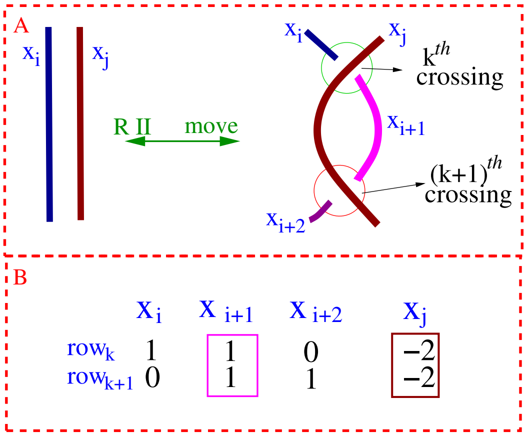
Figure 12: *Example: labeling crossings.*

Not all matrices that could correspond to a 3-string tangle are generated (see below). Not all generated matrices correspond to a tangle (see section on **Non-drawable matrices**). Many different matrices correspond to the same tangle (see below and section on **Equivalence moves**).

Matrices not generated. The algorithm under discussion does not generate all matrices which could correspond to a tangle. A tangle diagram can contain an extraneous crossing manifested by the looping of a string over itself. If the loop does not pass under any string, this results in the equation $x_i - x_{i+1} = 0$. This is more general than an RI move (Fig. 6) as there could be strings passing under this loop. In any case this tangle diagram can be simplified, and hence we do not need to generate the matrix corresponding to this diagram. Since all matrices generated have two "1"s and one "-2" in each row, none of the matrices generated will correspond to a tangle containing such an extraneous crossing.

Another case that is not generated is the presence of a string not crossing under any arcs, and hence consisting of just one arc doubly labeled x_i and x_{i+1}. This case results in the equation, $x_i - x_{i+1} = 0$. We could easily generate this, but the system of tangle equations in Fig. 4 rules out such tangles as possible solutions.

The algorithm also does not generate matrices that correspond to tangles containing crossings which can be removed by an RII move. These matrices contain -2's in the same column in two consecutive rows where the rows correspond to the same string. See Fig. 13. By not generating matrices containing the submatrix in Fig. 13B, we do not generate any tangle diagrams which can be simplified by an RII move (Fig.13). This also eliminates other tangles whose coloring matrix also contains this submatrix. This includes tangle diagrams containing a generalization of an RII move where strings are allowed to pass under the strings which would otherwise correspond to an RII move (Fig. 14, left-side) as well as tangles containing diagrams like that on the right-side of Fig. 14. All of these tangle diagrams can be simplified. This is one advantage of using coloring matrices to generate tangles. We easily remove a number of matrices that correspond to tangle diagrams where the number of crossings can be reduced.



$$\begin{array}{c|cccc} & x_i & x_{i+1} & x_{i+2} & x_j \\ \hline \text{row}_k & 1 & 1 & 0 & -2 \\ \text{row}_{k+1} & 0 & 1 & 1 & -2 \end{array}$$

Figure 13: *A. An RII move. B. Matrix corresponding to RII move.*

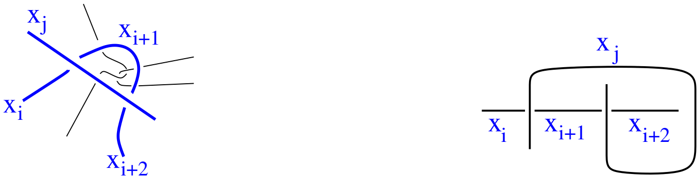


Figure 14: *Tangles which would also contain the submatrix in Fig. 13B.*

2-string tangle simplification

We can simplify the system of tangle equations in Fig. 4 by applying 2-string tangle analysis. By combining the 3-string cyan tangle and the 5-string green annulus tangle, we obtain a 2-string tangle. For example the tangles in Fig. 15 are all 2-string tangles (note 2-string tangles have four endpoints).

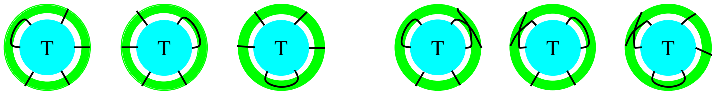


Figure 15: 2-string tangles from Fig. 4.

We can solve for the 2-string tangles in Fig. 15 using the tangle equations in Fig. 4. This step requires some mathematical background in tangle analysis although software available at KnotPlot.com does exist for solving some 2-string tangle equations [Darcy IK, Scharein, RG: TopoICE-R. Unpublished data]. For information on how to solve 2-string tangle equations, see [1, 17]. We can use a theorem in [18] and tangle calculus [1] to solve for one of these 2-string tangles (Fig. 16, where the crossings are either all right-handed or all left-handed):

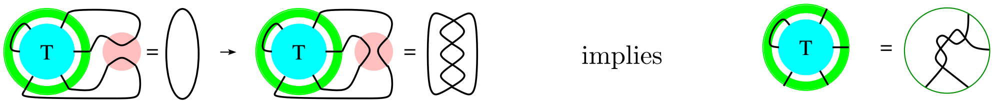


Figure 16: Solving for a 2-string tangle.

Similarly, by [19] and tangle calculus [1], we can solve for two more of these 2-string tangles (Fig. 17, where the crossings are either all right-handed or all left-handed):

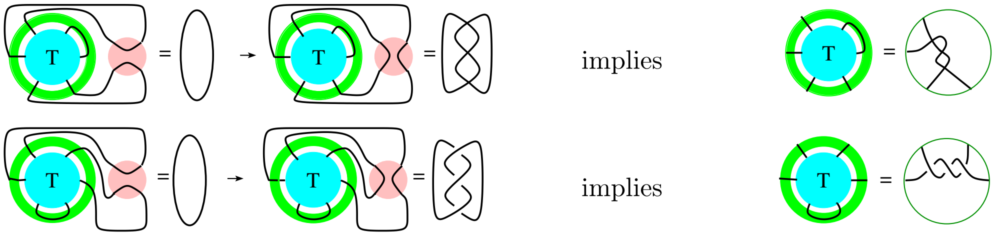


Figure 17: Solving for two more 2-string tangles.

This determines the remaining 2-string tangles in Fig. 15. In fact solving the system of tangle equations in Fig. 4 is equivalent to solving the system of three tangle equations in Fig. 18 for the 3-string tangle T.

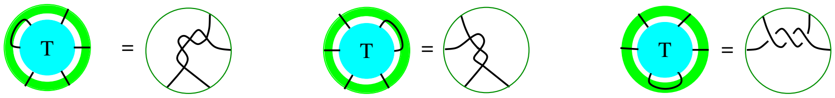


Figure 18: Tangle equations (crossings are either all right-handed or all left-handed).

Checking the coloring invariant

We first check if a generated matrix could be the coloring matrix of a tangle, $\mathbf{T}$, which satisfies the system of tangle equations in Fig. 18. In order to use the coloring invariants, $\mathbf{M_l}(\mathbf{T})$, $\mathbf{d_u}(\mathbf{T})$, we must first move the six columns corresponding to the endpoint arcs so that they become the six rightmost columns of the coloring matrix. For convenience, we will re-label these endpoint arcs as $x_1, x_2, ..., x_6$ as shown in Fig 19.

Given a 3-string tangle $\mathbf{T}$ with k crossings, let $\mathbf{M_T}$ be its $k \times (k+3)$ coloring matrix. Let $0_{3\times(k-3)}$ be a matrix with all zero entries. Suppose $SF(\mathbf{M_T})$ is as in equation 4:

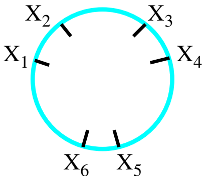


Figure 19: *Re-labeled endpoint arcs.*

$$SF(\mathbf{M_T}) = \left(\begin{array}{c|c} A_{(k-3)\times(k-3)} & B_{(k-3)\times 6} \\ \hline 0_{3\times(k-3)} & M_{3\times 6} \end{array}\right) \tag{4}$$

If **T** is a solution to the system of tangle equations in Fig. 18, then connecting the endpoint arcs, x_1 and x_2 of **T** results in the four crossing 2-string tangle $\mathbf{T_{12}}$ shown in Fig. 20. The coloring invariants of $\mathbf{T_{12}}$ are

$$\mathbf{d_u}(\mathbf{T_{12}}) = 1 \text{ and } \mathbf{M_l}(\mathbf{T_{12}}) = \begin{pmatrix} 1 & 0 & 4 & -5 \\ 0 & 1 & 3 & -4 \end{pmatrix} \text{ or } \begin{pmatrix} 1 & 0 & -4 & 3 \\ 0 & 1 & -5 & 4 \end{pmatrix}$$

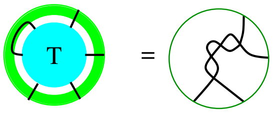


Figure 20: $\mathbf{T_{12}}$

Connecting endpoint arcs x_1 and x_2 of **T** to obtain the 2-string tangle $\mathbf{T_{12}}$ results in adding the equation $x_1 - x_2 = 0$ to the matrix $\mathbf{M_T}$ to obtain the matrix $\mathbf{M_{T_{12}}}$ (Eqn. 5).

$$\mathbf{M_{T_{12}}} = \left(\begin{array}{c|c} A_{(k-3)\times(k-3)} & B_{(k-3)\times 6} \\ \hline 0_{3\times(k-3)} & M_{3\times 6} \\ \hline 0_{1\times(k-3)} & 1\ -1\ 0\ 0\ 0\ 0 \end{array}\right) \tag{5}$$

If **T** is a solution to the tangle equation in Fig. 20, then $\mathbf{M_{T_{12}}}$ is a coloring matrix for $\mathbf{T_{12}}$. Let $\mathbf{M_{12}}$ be the 4×6 matrix obtained by adding the equation $x_1 - x_2 = 0$ to the matrix $M_{3\times 6}$ (Eqn. 6). Since $\mathbf{d_u}(\mathbf{T_{12}}) = 1$ and $\mathbf{M_l}(\mathbf{T_{12}}) = \begin{pmatrix} 1 & 0 & 4 & -5 \\ 0 & 1 & 3 & -4 \end{pmatrix}$ or $\begin{pmatrix} 1 & 0 & -4 & 3 \\ 0 & 1 & -5 & 4 \end{pmatrix}$, if **T** is a solution to the tangle equation in Fig. 20, then $det(A) = \pm 1$ and

$$M_{12} = \left(\begin{array}{c} M_{3\times 6} \\ \hline 1\ -1\ 0\ 0\ 0\ 0 \end{array}\right) \sim \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & * & * \\ 0 & 0 & 1 & 0 & 4 & -5 \\ 0 & 0 & 0 & 1 & 3 & -4 \end{pmatrix} \, or \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & * & * \\ 0 & 0 & 1 & 0 & -4 & 3 \\ 0 & 0 & 0 & 1 & -5 & 4 \end{pmatrix} \tag{6}$$

Hence, in order to determine if a matrix could correspond to a tangle, **T**, which is a solution to the tangle equation in Fig. 20, we need to check if $detA = \pm 1$ and if M_{12} is row equivalent to the 4×6 matrix in Eqn. 6. This is not a guarantee that **T** is a solution as different tangles can have the same coloring invariants (Navarra-Madsen, J. and Darcy, IK: Colorability and n-String Tangles, Unpublished data), but it is sufficient for solving the tangle equations in Fig. 18.

Similarly to determine if $\mathbf{T}$ could be a solution to the tangle equation in Fig. 21, we add the equation $x_3 - x_4 = 0$ to the matrix $\mathbf{M_T}$ and check if this matrix satisfies the coloring invariants of $\mathbf{T_{34}}$ as given in Eqn. 7.

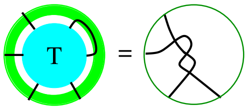


Figure 21: $\mathbf{T_{34}}$

$$\mathbf{d_u}(\mathbf{T_{34}}) = 1, \quad \mathbf{M_l}(\mathbf{T_{34}}) = \left(\begin{array}{cccc} 1 & 0 & 3 & -4 \\ 0 & 1 & 2 & -3 \end{array} \right) \text{ or } \left(\begin{array}{cccc} 1 & 0 & -3 & 2 \\ 0 & 1 & -4 & 3 \end{array} \right) \tag{7}$$

Finally, we determine if $\mathbf{T}$ could be a solution to the tangle equation in Fig. 22, by adding the equation $x_5 - x_6 = 0$ to the matrix $\mathbf{M_T}$ and checking if this matrix satisfies the coloring invariants of $\mathbf{T_{56}}$ as given in Eqn. 8.

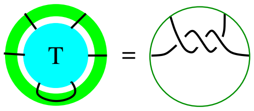


Figure 22: $\mathbf{T_{56}}$

$$\mathbf{d_u}(\mathbf{T_{56}}) = 1, \quad \mathbf{M_l}(\mathbf{T_{56}}) = \left(\begin{array}{cccc} 1 & 0 & 3 & -4 \\ 0 & 1 & 2 & -3 \end{array} \right) \tag{8}$$

Alternatively, we can determine what the entries of the submatrix $M_{3\times 6}$ of $\mathbf{M_T}$ (Eqn. 4) need to be in order for $\mathbf{T}$ to satisfy the tangle equations in Fig. 18. To determine $M_{3\times 6}$, we add the equations $x_i - x_{i+1}$ for each $i = 1, 3, 5$, and determine the constraints needed to satisfy the coloring invariants of $\mathbf{T_{i(i+1)}}$. If $\mathbf{T}$ satisfies the tangle equations in Fig 18, then the determinant of A, the upper $(k-3)\times(k-3)$ submatrix of $\mathbf{M_T}$ is ±1 and $M_{3\times 6}$ is as in Eqn. 9.

$$M_{3\times 6} \sim \begin{pmatrix} 1 & -1 & 1 & -1 & 1 & -1 \\ 0 & 1 & t & -1-t & s-r-x & -s+r+x \\ 0 & 0 & x & 1-x & r+x & -1-r-x \end{pmatrix} \tag{9}$$

for some integer x, where $r = 3$ or -5, $s = 2$ or -4, and $t = 2$.

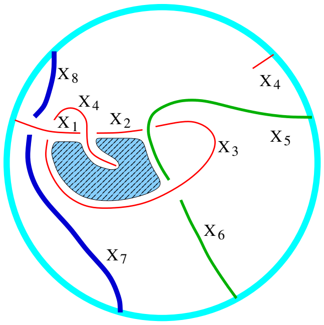
As a check, both methods were implemented.

Non-drawable matrices

Not all generated matrices correspond to a tangle. See for example, Fig. 23. If the matrix in Fig. 23 corresponds to a coloring matrix of a tangle, then since it has five rows, the tangle must have five crossings. Also, the first string should consist of four arcs, x_1, x_2, x_2, x_4, while the second string consists of arcs x_5, x_6 and the third string consists of arcs x_7 and x_8. If we can embed all the arcs so that the matrix corresponds to a coloring matrix of the resulting tangle, then the tangle corresponding to the matrix is drawable.

Note there is a -2 in the first row and fourth column of the matrix in Fig. 23. Recall the first row represents the underarcs x_1,x_2 while the fourth column represents the overarc x_4. Hence the arc x_4 must cross over between the arcs x_1 and x_2. Since x_4 is also an endpoint arc, it must also connect to the boundary of the 3-ball. However, after passing over between the arcs, x_1 and x_2, the arc x_4 arc is trapped in the shaded region and cannot connect to the boundary of the 3-ball without introducing an extra crossing. Thus this matrix does not correspond to a drawable tangle.

We use an algorithm similar to that described in [13] to completely determine if a matrix corresponds to a drawable tangle. This algorithm determines if all arcs can be drawn or if an arc becomes trapped in a region and cannot be completed.



$$\begin{pmatrix} 1 & 1 & 0 & -2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & -2 & 0 & 0 & 0 \\ -2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 1 & 1 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Figure 23: *A matrix that does not correspond to a 3-string tangle.*

Equivalence moves.

Recall that a tangle can be represented by a number of different diagrams related by Reidemeister moves. While generating matrices, we omit matrices where the corresponding diagram can be simplified by R1 or R2 moves and other matrix related moves (as described in section **Tangle generation**). We also added two additional equivalence relations.

We removed tangles containing the diagram shown in Fig. 24 by removing matrices containing the submatrices in Eqn. 10.

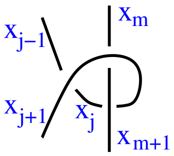


Figure 24: *A diagram corresponding to equation 10.*

$$\left(\begin{array}{c|ccc|cc} & x_{j-1} & x_j & x_{j+1} & x_m & x_{m\pm1} \\ \hline i & 1 & 1 & -2 & 0 & 0 \\ i+1 & 0 & 1 & 1 & 0 & -2 \\ \hline k & 0 & 0 & -2 & 1 & 1 \end{array}\right) \;\&\; \left(\begin{array}{c|ccc|cc} & x_{j-1} & x_j & x_{j+1} & x_m & x_{m\pm1} \\ \hline i-1 & 1 & 1 & 0 & 0 & -2 \\ i & -2 & 1 & 1 & 0 & 0 \\ \hline k & -2 & 0 & 0 & 1 & 1 \end{array}\right) \tag{10}$$

This also eliminates other tangle diagrams whose matrices contain these submatrices, but all such tangles can be simplified.

A tangle diagram containing the left-hand side of an RIII move (Fig. 6) will be equivalent to the tangle diagram obtained after the RIII move has been performed. Hence we choose one of these tangle diagrams and discard the other. After the above equivalence moves, we are left with thirteen possible tangles which can be checked by hand to determine if they correspond to equivalent or non-equivalent solutions to the tangle equations in Fig 4.

Authors contributions

CM and JN contributed to the mathematical analysis for applying the coloring invariant. JN also drafted significant portions of the sections **The coloring invariants, Tangle generation, Checking the coloring invariants**. AP, JS, TT developed the software implementing the coloring invariant calculations. RM contributed to the **Non-drawable** section and is responsible for the subroutine which determines if a matrix corresponds to a drawable tangle. He was assisted by ND and JS. JC, ND, SM, and JS developed equivalence moves which were implemented by ND, RM, and JS. ID conceived of and oversaw this project, drafted much of the manuscript, and contributed to the mathematical and software development. All authors read and approved the final manuscript.

Acknowledgements

This work was supported by a grant from the Joint DMS/NIGMS Initiative to Support Research in the Area of Mathematical Biology to I.D and S. D. Levene. (NIH GM 67242) and by an Interdisciplinary Research Grant from The University of Iowa's Obermann Center for Advanced Studies to I.D, S. D. Levene, and R. G. Scharein. We also thank John Luecke who ran a VIGRE REU at UT Austin which supported J.C., N.D., S.M., and J.S.. N.D. and J.S. were also supported through the U.I. Undergraduate Scholar Assistant (USA) Program.

We also thank Abhijeet Bhutra for his contributions to the software discussed in this paper and Rob Scharein and KnotPlot.com for figures 1A, 5, 6, and 9.

References

1. Ernst C, Sumners DW: **A calculus for rational tangles: applications to DNA recombination**. *Math. Proc. Camb. Phil. Soc.* 1990, **108**:489–515.
2. Crisona N, Weinberg R, Peter B, Sumners D, Cozzarelli N: **The topological mechanism of phage lambda integrase**. *J Mol Biol.* 1999, **289(4)**:747–75.
3. Vazquez M, Colloms SD, Sumners D: **Tangle analysis of Xer recombination reveals only three solutions, all consistent with a single three-dimensional topological pathway.** *J Mol Biol.* 2005, **346(2)**:493–504.
4. Vetcher AA, Lushnikov AY, Navarra-Madsen J, Scharein RG, Lyubchenko YL, Darcy IK, Levene SD: **DNA Topology and Geometry in Flp and Cre Recombination.** *J Mol Biol.* 2006.
5. Pathania S HR Jayaram M: **Path of DNA within the Mu transpososome. Transposase interactions bridging two Mu ends and the enhancer trap five DNA supercoils**. *Cell* 2002, **109(4)**:425–436.
6. Chaconas G, Harshey R: **Transposition of phage Mu DNA.** *In Mobile DNA II (eds. N.L. Craig et al.) ASM Press* 2002.
7. Grainge I, Buck D, Jayaram M: **Geometry of site-alignment during Int family recombination**. *J. Mol. Biol.* 2000, **298**:749–764.
8. Kilbride E, Boocock M, Stark W: **Topological selectivity of a hybrid site-specific recombination system with elements from Tn3 res/resolvase and bacteriophage PL lox$P$/Cre.** *J. Mol. Biol.* 1999, **289**:1219–1230.
9. Fox RH: **Metacyclic Invariants of Knots and Links**. *Canadian Journal Math* 1970, **22**:193–201.
10. Livingston C: *Knot Theory.* Washington, DC: Math. Assoc. Amer. 1993.
11. Dowker CH, Thistlethwaite MB: **Classification of Knot Projections.** *Topol. Appl.* 1983, **16**:19–31.
12. Ewing B, Millett KC: **Computational algorithms and the complexity of link polynomials.** *In Progress in Knot Theory and Related Topics, Trauaux en Cours, Hermann. Paris* 1996.
13. Doll H, Hoste J: **A tabulation of oriented links.** *Math. Comp.* 1991, **57**:747–761.
14. Yuan JF, Beniac DR, Chaconas G, Ottensmeyer FP: **3D reconstruction of the Mu transposase and the Type 1 transpososome: a structural framework for Mu DNA transposition .** *GENES & DEVELOPMENT* 2005, **19**:840–852.
15. Kilbride EA, Burke ME, Boocock M, Stark W: **Determinants of product topology in a hybrid Cre-Tn3 resolvase site-specific recombination system.** *J. Mol. Biol.* 2006, **355(2)**:185–95.
16. Bar-Natan D: **The Mathematica Package KnotTheory.** *http://katlas.math.toronto.edu/wiki/The_Mathematica_Package_KnotTheory*.
17. Darcy I: **Solving unoriented tangle equations involving 4-plats**. *J. Knot Theory Ramifications* 2005, **14**:993–1005.
18. Hirasawa M, Shimokawa K: **Dehn surgeries on strongly invertible knots which yield lens spaces**. *Proc. Amer. Math. Soc.* 2000, **128**:3445–3451.
19. Kronheimer P, Mrowka T, Ozsvath P, Szabo Z: **Monopoles and lens space surgeries.** *http://xxx.lanl.gov/find*. mathGT/0310164.

Figures

Figure 1

A.) Some 2-string tangles.   B.) a 3-string tangle.

Figure 2 - Difference topology experiment

Mu represented by the cyan colored ball is shown bound to five DNA crossings. Cre is represented by the smaller pink ball. Before Cre recombination, the DNA is circular and unknotted. Cre recombination changes the DNA configuration outside of the Mu transpososome. Since four of the five crossings bound by Mu are trapped by Cre recombination, the DNA product configuration equals a four crossing link.

Figure 3

Tangle model from [5].

Figure 4

Tangle equations corresponding to difference topology experiments in [5].

Figure 5

A three-branched tangle.

Figure 6

Reidemeister moves.

Figure 7

Coloring A 2-string Tangle.

Figure 8

Possible parities.

Figure 9

The two possible Fig. 10B parity solutions

Figure 10

Possible parities.

Figure 11

Example: labeling arcs.

Figure 12

Example: labeling crossings.

Figure 13

A. An RII move. B. Matrix corresponding to RII move.

Figure 14

Tangles which would also contain the submatrix in Fig. 13B.

Figure 15

2-string tangles from Fig. 4.

Figure 16

Solving for a 2-string tangle.

Figure 17

Solving for two more 2-string tangles.

Figure 18

Tangle equations (crossings are either all right-handed or all left-handed).

Figure 19

Re-labeled endpoint arcs.

Figure 20

$\mathbf{T_{12}}$

Figure 21

$\mathbf{T_{34}}$

Figure 22

$\mathbf{T_{56}}$

Figure 23

A matrix that does not correspond to a 3-string tangle.

Figure 24

A diagram corresponding to equation 10.

Tables

Table 1 - Results

Number of matrices with the correct coloring invariants (Col columns), corresponding to a drawable tangle (Draw columns), and which are potentially non-equivalent (Non-equiv columns). The first column refers to the number of crossings in the tangle diagram. The second column gives the number of matrices generated which could correspond to a tangle with a fixed crossing number. The results in the next three columns assume the parity in Fig. 8A while the results in the last three columns assume the parity in Fig. 8B. The columns labeled "Col" state the number of generated matrices which have the correct coloring invariants to satisfy the equations in Fig. 4. However, not all generated matrices correspond to a tangle. The columns labeled "Draw" give the number of matrices which correspond to a drawable tangle with the correct coloring invariants. The number of these matrices which may correspond to non-equivalent tangles is given in the columns labeled "Non-equiv?". Note, however, that the algorithm does not identify all equivalent tangles.

# of Crossings	# of Matrices Generated	Parity Fig 8A			Parity Fig. 8B		
		Col	Draw	Non-Equiv?	Col	Draw	Non-Equiv?
≤ 4	1,639	0	0	0	0	0	0
5	34,578	1	1	1	1	0	0
6	794,578	22	4	0	22	0	0
7	19,781,058	354	15	3	400	0	0
8	537,193,563	5019	106	6	5595	6	3