

THE MASTER ARRAY, A COMPLETE INVARIANT  
FOR PRIME ALTERNATING LINKS

(Spine title: The Master Array of a Prime Alternating Link)

(Thesis format: Monograph)

by

Ortho Flint aka D.K. Smith

Graduate Program  
in  
Mathematics

A thesis submitted in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy

Faculty of Graduate Studies  
The University of Western Ontario  
London, Ontario, Canada

© Ortho Flint 2007

THE UNIVERSITY OF WESTERN ONTARIO  
FACULTY OF GRADUATE STUDIES

**CERTIFICATE OF EXAMINATION**

Supervisor

\_\_\_\_\_  
Dr. Stuart Rankin

Supervisory Committee

\_\_\_\_\_  
Dr. J. Dan Christensen

\_\_\_\_\_  
Dr. Graham Denham

Examiners

\_\_\_\_\_  
Dr. W. B. Raymond Lickorish

\_\_\_\_\_  
Dr. Graham Denham

\_\_\_\_\_  
Dr. Richard Kane

\_\_\_\_\_  
Dr. Mark Daley

The thesis by

**Ortho Flint**

entitled:

**The Master Array, a Complete Invariant for Prime Alternating Links**

is accepted in partial fulfillment of the  
requirements for the degree of  
Doctor of Philosophy

Date \_\_\_\_\_

\_\_\_\_\_  
Chair of the Thesis Examination Board

## Abstract

This dissertation describes in full the construction for any prime alternating link  $L$  of an integer array, called the *master array* for  $L$ , from any reduced alternating diagram for  $L$ . The master array has the property that two links are equivalent up to mirror image if and only if their master arrays are identical; that is, up to mirror image, the master array is a complete invariant for prime alternating links. Furthermore, every reduced alternating diagram of a prime alternating link  $L$  can be constructed from the master array for  $L$ .

The existence of this invariant relies equally on the Tait flyping conjecture and the notion of non-interference of flype orbits.

From a reduced alternating diagram for a prime alternating link  $L$ , a condensed form of gauss code, which we call a *group code* for  $L$  is first constructed. This code is then augmented with flype information for every crossing of the diagram, and the result is called a *master group code* for  $L$ . To obtain the master array for  $L$ , a procedure which we call the *standardization* procedure is applied to the constructed master group code, with the result independent of the particular master group code.

The master group code construction relies on a surgery that we call the  $\mathcal{D}$  surgery. This surgery replaces a tangle consisting of a single crossing by a tangle consisting of a 2-group; that is, a tangle consisting of two crossings with two arcs in common.

Furthermore, we establish an upper bound on the size of the master array of a prime alternating link  $L$  in terms of the number of crossings in a reduced alternating diagram for  $L$ , and we give a family of prime alternating links that achieve this bound, so this bound is best possible.

The master array proved to be pinnacle for the enumeration of all unoriented and oriented prime alternating links up to 23 crossings inclusive (at which point, we had used three terabytes of disk storage and so we decided to stop), not only for the assurance that all links in the tables are distinct but for the huge reduction of time and resources over all previously known enumeration methods.

**Keywords:** Complete invariant, master array, master group code, prime, alternating, link, knot, group code, gauss code, Dowker-Thistlethwaite code, knot equivalence, enumeration, flyping, knot diagram, tangle, surgery

## Epigraph

Beware the incarcerated man, for it is his mind that liberates itself which is the genesis of all good and evil that befalls his kind.

## Dedication

To my wife Ghia and our son Colton

## Acknowledgements

I would like to thank my supervisor, Professor Stuart Alexander Rankin, for his many years of tutelage and unbounded patience. After my supervisor, the next three most important research colleagues that I would like to thank are (in the chronological order in which we worked together) John Schermann, Peter de Vries, and Bruce Fontaine. Finally, I would like to thank the Department of Mathematics and the Department of Applied Mathematics of the University of Western Ontario for a nurturing atmosphere, their moral and financial support, and for giving me the chance to realize my desires.

## TABLE OF CONTENTS

Abstract . . . . .	iii
Acknowledgements . . . . .	vi
I. Introduction and preliminaries . . . . .	1
1. Conventions . . . . .	1
2. Early history . . . . .	2
3. A paradigm shift . . . . .	5
4. The end of hand tabulation . . . . .	6
5. The modern era . . . . .	7
II. Link encoding . . . . .	14
III. Flype orbits . . . . .	17
1. The orbit of a group . . . . .	17
2. Flype orbit tangles . . . . .	31
IV. Master group code . . . . .	35
1. Converting a link diagram to a knot diagram . . . . .	35
2. Master group code . . . . .	41
3. An algorithm for the construction of a master group code . . . . .	44
4. The construction of all diagrams for a link . . . . .	57
V. The master array . . . . .	59
1. The construction of the master array from a master group code . . . . .	59
2. Bounds on the size of master group codes and the number of diagrams . . . . .	70
VI. Conclusions . . . . .	76
1. Complexity . . . . .	76
2. Enumeration . . . . .	77
3. Composite links . . . . .	77
Bibliography . . . . .	78
Vita . . . . .	80

## CHAPTER I

### INTRODUCTION AND PRELIMINARIES

#### 1. Conventions

A link  $L$  of  $m$  components is a subset of  $S^3$  that consists of  $m$  disjoint, piecewise linear simple closed curves, each of which is said to be a component of the link. A link of one component is a knot.

A diagram  $D$  of a link  $L$  is the image of  $L$  under a piecewise linear projection mapping  $\pi$  of  $S^3$  onto a 2-sphere disjoint from  $L$  such that for every  $p \in D = \pi(L)$ ,  $|\pi^{-1}(p) \cap L| \leq 2$  and if  $p \in D$  is such that  $\pi^{-1}(p) \cap L = \{x_1, x_2\}$  with  $x_1 \neq x_2$ , then neither  $x_1$  nor  $x_2$  is an endpoint of a linear segment of  $L$ . A point  $p \in D$  such that  $|\pi^{-1}(p) \cap L| = 2$  is said to be a crossing of  $D$ , called a component crossing if both points of  $\pi^{-1}(p) \cap L$  belong to the same component, otherwise it is called a link crossing.

Links  $L_1$  and  $L_2$  in  $S^3$  are equivalent if there is an orientation-preserving homeomorphism  $h: S^3 \rightarrow S^3$  such that  $h(L_1) = L_2$ . A diagram for a link  $L_1$  is said to be equivalent to a diagram for a link  $L_2$  if  $L_1$  is equivalent to  $L_2$ .

It is conventional to indicate on a link diagram the additional information required to construct a link equivalent to the one from which the diagram was made. This information takes the form of over/under passes, indicated by a break in the image of the underpassing arc in a neighbourhood of the crossing.

We shall treat a link diagram as a 4-regular graph on  $S^2$  by considering each crossing as a vertex of the graph and the portion of the curve between two consecutive crossings as an edge between the two vertices. There is a natural cyclic ordering of the four edges incident to a vertex obtained by visiting the edges in a clockwise order. We shall say that two edges with an endpoint  $v$  in common are adjacent if they appear consecutively in the cyclic list of edges incident to  $v$  (note this is

more restrictive than the usual use of the word adjacent). In a link diagram, a circuit in which no two adjacent edges appear consecutively in the circuit will visit all vertices in some component and no others, and we shall call such a circuit a traversal of that component. The traversal is said to be alternating if consecutive crossings alternate between underpass and overpass. A link traversal is a set of component traversals containing exactly one component traversal of each component. An alternating diagram is a link diagram for which every link traversal is alternating. An alternating link is one for which at least one diagram is alternating.

For a link  $L$ , we consider the set  $Diag(L)$  of all link diagrams of all links equivalent to  $L$ . A diagram of fewest crossings in  $Diag(L)$  is called a minimal diagram for  $L$ .

## 2. Early history

The desire to enumerate knots and links has provided the incentive for all knot encoding schemes. In each case, a scheme for encoding a knot diagram is given, with the idea that from the encoding, a diagram for the link could be reconstructed.

The history of knot encoding begins with K.F. Gauss. Gauss tried to classify closed plane curves with a finite number of self-intersections, with no point on the curve having multiplicity greater than two, by means of what we now call a gauss code for such a curve. Gauss made a code of such a curve by assigning letters to the crossing points (points where the curve intersects itself) and then used this labelling to formulate a sequence of these labels. The modern approach is to use the integers 1 to  $n$  as labels for a curve with  $n$  points of self-intersection.

The notion of gauss code is easily extended to links, and we describe here a method for generating a gauss code for a given link diagram. Given a diagram for a link, we choose any component to start with and select any point  $p$  on the component's image in the diagram, subject to the requirement that  $p$  not be a crossing. Choose a direction to traverse the component, starting at  $p$ , and as each unlabelled crossing is encountered, write down a label distinct from any label used so far. When the component has been completely traversed, select another component and repeat the process. When all crossings have been labelled, a gauss code for the diagram is constructed by traversing every component of the link in an arbitrary order, and with each component traversal having arbitrary orientation, starting at the reference point  $p$  for that component and record the crossing labels encountered in order of traversal. If traversing a crossing on an under-passing strand, then record the label of the crossing with a negative sign. The procedure terminates when all crossings

in each component have been traversed (necessarily, each label will appear twice in the code, once for the over-passing arc and once for the under-passing arc). It is customary to place a colon between consecutive component label sequences. Further, observe that the sequence of labels for each component is cyclic, in that any cyclic permutation of the labels simply corresponds to a different choice of reference point on the component.

Gauss had worked out a set of rules for a sequence of positive integers to be admissible as a gauss code for such a curve. These rules did not in fact characterize gauss codes, and Gauss found that while he was able to construct all gauss codes for knots of at most 4 crossings, his rules were inadequate for the task of enumerating the knots of 5 crossings and he had to leave the job unfinished (a characterization of gauss code was finally provided in 1968 by L. P. Treybig [27]). At this point, a student of Gauss, J. H. Listing, became interested in the problem, presenting some very important ideas in a publication [12] in 1847. At about the same time, and unaware of Listing's work, P. G. Tait began to deliberate on the problem of knot classification. He realized that when one creates a diagram from a gauss code for a prime link, the result is either equivalent to the diagram from which the code was made or to its mirror. He developed the extended gauss code (and another encoding scheme, which is essentially the code that is today referred to as the *DT* code), wherein he included both writhe and under/over-pass information, where writhe is defined as follows.

**I.2.1 DEFINITION.** Given an orientation on each component of a link diagram  $D$ , the *writhe* of a crossing is either  $+1$  or  $-1$  according to the criteria shown in Figure 1.



Figure 1. The writhe of a crossing

Now, with the writhe for each crossing taken into consideration, the resulting diagram will be equivalent to the diagram from which the code was made. Appending the writhe for each crossing to the gauss code gives us the *extended gauss* code. One way to append the writhe information is as follows:

The crossings are labelled the same as before and the crossings are recorded as before except that rules for assigning the signs are revised slightly. The first time a crossing's label is encountered in the prescribed traversal, then as before, the over-pass is unsigned while an under-pass is negative. The second time a crossing's label is encountered in the code, its label is to be assigned a sign depending on the handedness

of the crossing. If it is right-handed (a positive writhe), the label is left without a sign, while if it is left-handed (a negative writhe), the label is given a minus sign.

Tait clearly understood the notion of a *flype* move (which he called a twist move), and its importance to the problem of distinguishing between diagrams which represent the same alternating knot, and accordingly, the famous Tait flyping conjecture (not actually formulated explicitly by him) bears his name today. Using extended gauss code and his twist moves, Tait was able to construct a table of knots of crossing number seven. As a result of the difficulty of the task, he came to believe that a tabulation of higher crossing knots would require either more efficient methods or some mechanized approach for the determination of whether or not two knot diagrams were the same, an idea that would not be realized for over 100 years. Consequently, between 1877 and 1883, Tait did little work in knot theory. However, during this time he became aware of Listing's work and its importance to his work on knots, and in late 1883, he presented an address on Listing's work on topology to the Edinburgh Mathematical Society, in which he mentioned the problem of enumerating knots and how he now felt that it would be "a mere question of skilled labour". Subsequently, Tait was contacted by Reverend Thomas Penyngton Kirkman, Rector of Croft, Lancashire, who had spent a significant portion of the previous 30 years considering combinatorial problems involving graphs and hypergraphs. Kirkman viewed the knot enumeration problem as a problem of enumerating particular 4-regular planar graphs that could be projections of knots or links and set to work on the problem. The first result that appeared from this collaboration was Tait's paper [24] in which he presented their enumeration of the knots of at most 9 crossings, followed by [25] in which he described his work with Kirkman to produce the enumeration of the ten crossing prime alternating knots. While Tait was not completely mathematically satisfied with the tables in that he had no proof of the inequivalence of each of the knots (lending support to the view that he did not take his flyping conjecture to be true), he was satisfied that the list was complete. As Tait prepared to publish the 10 crossing prime alternating knot enumeration, he received an enumeration of knots up to 10 crossings from Charles N. Little, a mathematics Ph. D. student of H. A. Newton at Yale University. (His knot tabulation served as his doctoral dissertation, entitled "On Knots, with a Census for Order 10", which earned him his doctorate in 1885.) Comparing his list with Little's, Tait found one duplication in his list and a duplication and omission in Little's. After correcting his errors, he sent the paper to press. Before publishing the table up to 10 crossings, Tait had received a list of 1581 knot projections of 11-crossing knots from Kirkman, but he decided that the work

involved in determining the equivalences of the alternating knots with those projections was too great. Instead, he suggested that Little make an investigation of the 11-crossing knots. In [14], Little presented an overview of his enumeration methods for the 11 crossing prime alternating knots, and gave knot diagrams for 357 prime alternating knots (it was later to be learned that this list contained one duplicate and had eleven omissions). At the time that Tait, Kirkman, and Little were doing their enumerations, there was no necessary and sufficient condition for a sequence of positive integers to be a gauss code, nor was there a characterization of the 4-regular simple connected planar graphs, so they had no way of verifying that their lists were complete. Similarly, they did not have access to all of the tools that are available today for the determination of equivalence of diagrams, so it was inevitable that the lists would be incomplete and would contain duplicates.

Following the work of Tait, Kirkman, and Little, knot enumeration came to a virtual standstill for many decades. During this period, many new topological and algebraic tools were developed and used to rigorously show that the knots in the tables were truly different.

### 3. A paradigm shift

In 1967, J.H. Conway [3] introduced a fundamentally new idea. After describing the basic concept of a tangle, a portion of a knot diagram with four free-end strands, he gave a concise geometrical notation for describing knots and links in terms of their construction from tangles and polyhedra. Conway remarks that this construction can be traced back to ideas of T.P. Kirkman [10]. Although the notion of a tangle was implicit in Tait's work, Conway brought it into sharp focus, and today, the tangle concept is arguably one of the most important notions in all of knot theory.

Basically, Conway's notational system captures the relationship amongst the tangles that form a diagram. The Conway code for a link depends on having an enumeration of 4-regular simple connected planar graphs, and a vertex labelling for each using consecutive positive integers beginning at 1, and for each vertex in the graph, a labelling of the four edges incident to the vertex, using the labels  $NE, SE, SW, NW$  in either clockwise or counterclockwise order. With this list in hand, the notation for a particular link diagram consists of the name of the labelled graph in the list, followed by a sequence of specifications for *algebraic tangles*, one for each vertex in the graph, with the understanding that the  $i^{\text{th}}$  tangle in the sequence is to be placed at the vertex labelled  $i$  in such a way that its four edges, which carry the labels

$NE, SE, SW, NW$  in a clockwise fashion, are matched up with the location of the corresponding labels in the diagram. Since it is not our intention to discuss this encoding at length, suffice it to say that an algebraic tangle can be specified by a simple expression of an arithmetic nature. Conway did not give a convention for naming the graphs (each with its arbitrary but specified labelling) in this list, but he did enumerate, label, and assign a name to every 4-regular simple connected planar graph of at most 11 vertices. With these, he was able to enumerate all prime knots of at most 11 crossings and all links of at most 10 crossings. Conway's enumeration added eleven new prime alternating knots to the list of 11 crossing prime alternating knots that had been compiled by Little, and removed one duplicate from Little's list.

Conway did succeed in enumerating all 10 crossing prime knots, although his list contained two different diagrams of a prime nonalternating knot. This duplication was finally discovered in 1974 by K. A. Perko [17].

While Conway's notation was a conceptual breakthrough and is minimalist for knots of low crossing number, it was not effective for enumeration due to the huge increase in the number of constructed diagrams that need to be checked for duplication.

Hand tabulation came to an end at that time after collaborative work by F. Bonahon and L. Siebenmann and independent work by Perko [18] completed the tabulation of 11-crossing knots. C. H. Dowker and M. B. Thistlethwaite were the first to computerize knot enumeration, and their work [5] in the early 1980's brought the table of knots up to 13 crossings.

#### 4. The end of hand tabulation

In [5], Dowker and Thistlethwaite established a method to tabulate the diagrams of the prime knots of a given crossing size. In this paper, they remark that they used a modified version of an encoding scheme used by Tait [23], and that they had methods which used a collection of invariants to remove duplicates, resulting in a list of all knots up to 12 crossings. Their modification to Tait's encoding scheme was essentially to drop the writhe information and it is this modified version that we now call the  $DT$  code.

Here is a description of a method to construct a  $DT$  code for a knot diagram. Choose an arbitrary starting point and orientation and traverse the knot diagram, counting every crossing that is encountered. Given that every crossing is visited twice, then if the diagram has  $n$  crossings, the count ends at  $2n$ . Label each crossing with

the values the counter held when the crossing was visited, though when labeling by an even number, take it with a minus sign if you are traversing an over-passing arc of the crossing.

Every crossing is now labelled with two integers, one even and one odd, whose absolute values run from 1 to  $2n$ . The *DT* code of  $K$  is the sequence whose  $i^{\text{th}}$  entry is the even integer that was paired with the odd integer  $2i - 1$ ,  $i = 1, 2, \dots, n$ .

Since a *DT* code does not record the writhe of any crossing in the diagram, one might construct from a *DT* code the diagram of the mirror image of the knot from which the *DT* code was obtained. It is precisely this information that was removed by Dowker and Thistlethwaite from Tait's original coding scheme.

It was observed by Doll and Hoste [4] that the notion of a *DT* code can be extended to links. Follow the same numbering process as for knots, except when you finish traversing one component, move onto another component. There is always a choice of starting points along the components for which the resulting pairing is a pairing between odd and even numbers. The odd integers from 1 to  $2n - 1$  are partitioned by component, and arranged in increasing order within each component, with components listed in an arbitrary order. The sequences of even integers that are paired with the odd integers for a component are separated by a vertical dash.

After the work of Dowker and Thistlethwaite in the early 1980's, work halted again until the early 1990's, at which time a group of high-school students won time on a Cray supercomputer, and working under the guidance of Jim Hoste, were able to enumerate all alternating knots through 14 crossings [7] in two weeks time.

The next advance in enumeration was a major computational achievement by Hoste, Thistlethwaite, and Jeff Weeks, where in [8] they discuss the techniques they used to complete the tabulation of all knots through 16 crossings, finding 1, 701, 936 distinct prime knots of at most 16 crossings. Their work had a built-in check, as Hoste and Weeks worked together using hyperbolic invariants and Thistlethwaite worked independently using absolutely no hyperbolic invariants. The starting point for each crossing size was a list of all possible *DT* codes for knots of that crossing size (by now, the important criteria for a sequence of integers to be a gauss/*DT* code was available), and the independent teams used different means for the determination of equivalent codes (codes that represented different diagrams of the same knot). This work relied heavily on the Tait flying conjecture, due to Menasco and Thistlethwaite and which appeared in print in 1993 (see [15]).

## 5. The modern era

In 1997, a new approach to enumeration appears in the literature. In [2], Jorge Calvo introduces an iterative “bottom up” approach (starting with the minimal diagram of the unknot) for the enumeration of the prime alternating knots. Using a surgery he called *twisted splices*, he added new crossings to an alternating diagram to construct an alternating diagram of a knot of larger crossing size. This approach eliminated the task of creating and sorting *DT* codes, however the problem of identifying equivalent diagrams remains, and it is evident that there will be a similar explosion in the number of diagrams created, far in excess of the actual number of prime alternating knots at the given crossing size.

Calvo acknowledges the computational difficulties, and in fact, did not actually formulate an algorithm by which the enumeration might be carried out. However, in that paper, Calvo described the notion of what we have called a crossing’s flype orbit (while it is evident that earlier authors were aware of this notion, none seems to have presented it as explicitly as Calvo), the identification of the tangles over which a crossing may flype. Additionally, he noted that two crossings that had two arcs in common would have to have exactly the same flype orbit. Finally, and this is perhaps the most important observation, he saw that any two crossings which do not have the same flype orbit can be freely flyped to any of their respective flype positions independently of the other’s flyping activity. We refer to this as the non-interference of flype orbits, and this concept plays a fundamental role in our formulation of a complete invariant for prime alternating links.

Although Calvo had formulated key concepts, he did not have such a complete invariant to enable him to deal with the huge amount of redundancy in the set of diagrams that his techniques would produce. The enormity of this task was earlier mentioned in [8], wherein Hoste, Thistlethwaite and Weeks stated “Indeed, it is the task of grouping the diagrams together by knot type (which is by flype equivalence classes), rather than enumerating all possible diagrams that remains to this day the most difficult part of knot tabulation”.

This is precisely the problem we resolved completely for prime alternating knots [19], [20], and now for prime alternating links! By developing a new knot encoding scheme (extended to links in this thesis), we were able to develop both an algorithm for the generation of all prime alternating knots (and now links) and a complete invariant that could be effectively computed from any alternating diagram for a knot,

with which we could guarantee that our enumeration contained exactly one code for each distinct prime alternating knot.

We shall outline how we achieve this by first describing what it is we actually enumerate. In 1998, we developed a new knot encoding scheme, which was a useful variant of gauss code, which we called *group code*. This concept subsequently appeared in [19]. Our notation was originally designed specifically for encoding reduced alternating knot diagrams (that is, alternating diagrams without loops).

Briefly, the definition of group code as it appeared in [19] is described below along with the obvious extensions to incorporate writhe and under/over-pass information.

The definition of group code relies on the following concepts. Given any diagram  $D$ , a  $k$ -group, or a group of size  $k$ , is a sequence  $c_1, c_2, \dots, c_k$  of crossings, all of the same writhe, maximal subject to the requirement that for each  $i$  from 1 to  $k - 1$ ,  $c_i$  and  $c_{i+1}$  are joined by two strands. The two strands (or arcs) that are incident to  $c_1$  but not incident to  $c_2$  are called the strands at one end of the group, as are the two strands incident to  $c_k$  but not to  $c_{k-1}$ .

A portion of a traversal that begins at  $c_1$  and proceeds in the direction of  $c_2$ , stopping at  $c_k$ , is called an arc of the group, and each group has two group arcs. By a group of the diagram, we mean a  $k$ -group for some  $k \geq 1$ . A 1-group shall usually be referred to as a loner.

Let  $D$  denote a diagram of a knot  $K$ . For each  $k \geq 1$  for which  $D$  has at least one group of  $k$ , let  $i_k$  be the number of  $k$ -groups. For each such  $k$ , label the  $k$ -groups in an arbitrary fashion with the labels  $k_1, k_2, \dots, k_{i_k}$ . Assign an orientation to  $K$ . Then arbitrarily choose a group. Select an edge at one end of this group and begin a traversal of the diagram by following the selected edge into the group. Form the sequence of group labels in the order that the groups are encountered during this traversal. Note that each group label will appear exactly twice in this sequence.

For each group label  $k_i$ ,  $k > 1$ , that appears in this sequence, we check to see if both arcs of the group were traversed in the same direction (this check is meaningless when  $k = 1$ ), and if so, then the group is said to be a positive group, otherwise it is said to be a negative group and in this latter case, we place a minus sign in front of both occurrences of the group's label. In such a case, we think of the label as being of the form  $(-k)_i$ , and refer to the group as a group of size  $-k$ . In the case of a loner, the group labels will carry no sign.

For each group  $k_i$ , with  $|k_i| \geq 1$ , we now augment exactly one of the two group labels for  $k_i$  with the symbol  $(-)$  if the first crossing encountered in the group strand when following the traversal is an under-pass.

We remark that for a loner or a positive group or a negative group of odd size, exactly one group arc label will have  $(-)$  appended, while for a negative group of even size, either both group arcs or else neither group arc will have  $(-)$  appended.

Once the negative sign  $(-)$  assignments have been appended to the appropriate group labels, the result is a group code. Note that every group label (ignoring any appended  $(-)$ ) appears exactly twice in the group code.

As with a gauss code, a diagram constructed from a group code for a knot will either be equivalent to the diagram from which the code was constructed or to its mirror. If we take the writhe of each crossing into account, then from the code, we can construct a diagram known to be equivalent to the original. We explain below a method by which we may modify a group code to incorporate the writhe information. This modified group code shall be called the *extended group code*.

The groups are labelled the same as before except that any appended  $(-)$  that appears on the second appearance of a group arc label is removed. Rather, the second time a group's label is encountered in the code, we append  $(-)$  to the label if the group has negative writhe.

If we are encoding reduced alternating diagrams, then we can streamline gauss code, extended gauss code, group code, and extended group code as follows. In all cases remove sign augmentations that identify under/over-pass information. The resulting codes are called *alternating gauss codes*, *extended alternating gauss*, *alternating group*, and *extended alternating group* codes, respectively.

We provide examples of each encoding scheme described so far using the reduced alternating diagram shown in Figure 1. Since this is a reduced alternating diagram, we may begin labelling on an over-pass, thereby ensuring that no minus signs need be used in the *DT* code. We label the crossings as shown in Figure 1, and as a result, we find that  $[6, 10, 8, 2, 4]$  is a *DT* code for the diagram (and its mirror).

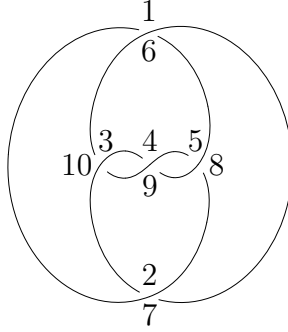


Figure 1. Diagram rendered by Knotilus [11]

With the same placement of the labels 1 through 5, we may construct an alternating gauss code, gauss code, alternating group code, and group code, respectively, for the diagram in Figure 1:

$$\begin{aligned}
 \text{alternating gauss code: } & 1, 2, 3, 4, 5, 1, 2, 5, 4, 3 \\
 \text{gauss code: } & 1, -2, 3, -4, 5, -1, 2, -5, 4, -3 \\
 \text{alternating group code: } & 2_1, -3_1, 2_1, -3_1 \\
 \text{group code: } & 2_1, -3_1, 2_1(-), -3_1(-).
 \end{aligned}$$

The following extended alternating gauss code, extended gauss code, extended alternating group code, and extended group code respectively, encode the mirror of the diagram shown in Figure 1:

$$\begin{aligned}
 \text{extended alternating gauss code: } & 1, 2, 3, 4, 5, -1, -2, -5, -4, -3 \\
 \text{extended gauss code: } & -1, 2, -3, 4, -5, -1, -2, -5, -4, -3 \\
 \text{extended alternating group code: } & 2_1, -3_1, 2_1(-), -3_1(-) \\
 \text{extended group code: } & 2_1(-), -3_1(-), 2_1(-), -3_1(-).
 \end{aligned}$$

Note that the group code and the extended alternating group code above are the same but the appended  $(-)$  denote different properties for the diagrams. Clearly, with all these variations it is important to know exactly which of the codes was constructed. For example, the left-handed trefoil can have a group code, extended alternating group code and a extended group code that is  $3_1, 3_1(-)$ , for its only reduced alternating diagram.

We remark that for diagrams of  $n$  crossings, a gauss code has length  $2n$ , a  $DT$  code has length  $n$ , while group codes can range in length from a minimum of 2 (the diagram consists of a single group, which is the case if and only if the link is a reduced alternating diagram for the  $n$ -crossing torus link) up to a maximum of  $2n$  (which occurs if and only if every group in the diagram is a loner, so the group code is a gauss code).

With the definition of group code in hand, the next step was to incorporate flype information into the knot encoding scheme. As described in [20], given an alternating

group code (which we shall call just a group code henceforth) for a minimal diagram for a prime alternating knot, we parse the code for each group in the code to determine all of the group's flype positions. During this parsing process, we may discover that crossings belonging to the same group; that is, those sharing the same flype orbit, are lying in different flype positions in the flype orbit. We call a group whose crossings are lying in different flype positions a *split* group. By parsing the complete orbit for a given group, we know all the crossings that form the group and we know the size of the *full* group which has this orbit. We then place a group label for this full group on each of the two arcs (or strands) that constitute a flype position for the group. All strands between connecting group arc are represented by the commas between group labels in the code, so we open the code at the appropriate place to insert a copy of the group's label at each flype position, and we append an index (for each group, the index is a non-negative integer, with 0 being the first index assigned) to the two arc's of one flype position to distinguish this position from all others. As a result of the non-interference of flype orbits, we may process the groups in any order whatsoever.

After every group in the original group code has had its flype orbit determined, and any split groups have been replaced by full groups, with all flype positions identified as described above, the resulting code is called a *master group code* for the knot. A master group code captures all the flying information of the alternating knot (as well as its mirror), and is calculated in linear time (in the number of crossings). Moreover, from any master group code for a prime alternating knot, we may obtain a group code for each and every minimal diagram of the knot.

In general, there will be more than one master group code for a prime alternating knot with respect to the labelling of the full groups of a given size, the indexing of the flype positions, and the group and direction from which we chose to write out the code. To remedy this, we introduced [20] a convention that would select from all master group codes a uniquely determined one, and that one we called the *master array* of the knot. Two prime alternating knots are then equivalent if and only if they have identical master arrays, and so the master array is, up to mirror image, a *complete invariant*.

Of course, it must be acknowledged that at the present time, there is no effective algorithm that will determine whether a given group, gauss, or *DT* code is a code for an alternating knot, and our computation of the master array depends on having a reduced alternating diagram of the knot. This should not be regarded as a limitation of the master array. The master array has proven to be an extremely valuable concept,

and in fact, it played a fundamental role in our enumeration of all prime alternating knots up at most 23 crossings (see [22] A002864, [20], and Knotilus [11]).

It should be clear that, due to the documentation of flype positions, the length of a master group code for the knot will usually be greater than that of the group code from which it was formed. In this dissertation, we extend the notion of master array to alternating links, and we prove that the length of any master array for an alternating link on  $n$  crossings is less than or equal to  $4(n-3)$ . We also establish  $2^{n-4}$  as an upper bound for the number of minimal diagrams (split or full group) for any prime alternating link on  $n$  crossings, and we provide a family of links that attains this bound.

The last historical comment to make, which may be interesting, if not at least a little curious, is that all the initial research (which led to the successful enumeration of the 18 and 19 crossing prime alternating knots in 1999) was created without any knowledge of the research work in knot theory from any era. For example, our term for a flype move was the Gross 3-component move, and it was a great relief when we learned in 2000 that our feeling that this was the only move that was required to establish equivalence of diagrams was actually known as the Tait flying conjecture and had been proven seven years before we had embarked on our research program. It is a certainty that had we been aware of others work, especially that of Thistlethwaite, we would not have had any incentive to choose the path we did, since this path would have appeared to be already well trodden.

We close this introduction with the remark that Tait's work on alternating knots implicitly relied on what has come to be called the first Tait conjecture; namely a reduced (loop free) alternating diagram of a link  $L$  is a minimal diagram for  $L$ . It was proven in 1987 by K. Murasugi [16], and shortly thereafter by L. Kauffman [9] and M. Thistlethwaite [26], that a diagram of an alternating link is minimal if and only if it is a reduced alternating diagram. In the work that follows, we shall be working only with prime alternating links. It will be IMPORTANT for the reader to note that for brevity, we shall hereafter use link to mean prime alternating link, and diagram of a link (prime alternating) to mean minimal diagram; that is, reduced and alternating.

## CHAPTER II

### LINK ENCODING

We introduce a new scheme for encoding an alternating link diagram. Such an encoding will be called a group code for the diagram.

**II.1 DEFINITION.** Given a link diagram, a  $k$ -group, or a group of size  $k$ , is a maximal length sequence  $c_1, c_2, \dots, c_k$  of crossings, subject to the requirement that  $k \geq 2$  and for each  $i$  from 1 to  $k - 1$ ,  $c_i$  and  $c_{i+1}$  are joined by two edges. The two edges that are incident to  $c_1$  but not incident to  $c_2$  are called the edges at one end of the group, as are the two edges incident to  $c_k$  but not to  $c_{k-1}$ . If  $c_1, c_2, \dots, c_k$  is a group, any subsequence of the form  $c_i, c_2, \dots, c_j$ , where  $1 \leq i < j \leq k$  is called a subgroup of the group  $c_1, c_2, \dots, c_k$ . More precisely, if  $t = j - i + 1$ , then  $c_i, c_2, \dots, c_j$  is called a  $t$ -subgroup of the  $k$ -group  $c_1, c_2, \dots, c_k$ . The two edges incident to  $c_i$  but not to  $c_{i+1}$  are called the edges at one end of the subgroup, as are the two edges that are incident to  $c_j$  but not to  $c_{j-1}$ .

A portion of a component traversal that begins at  $c_1$  and proceeds in the direction of  $c_2$ , stopping at  $c_k$ , is called an arc of the group. Each group has two group arcs and if both arcs belong to the same component, then we call the group a component group, otherwise we call the group a link group.

By a group of the diagram we mean a  $k$ -group for some  $k \geq 1$ . If  $G$  is a  $k$ -group, then we shall write  $|G| = k$ .

Schematically, we shall illustrate a group  $G$  by

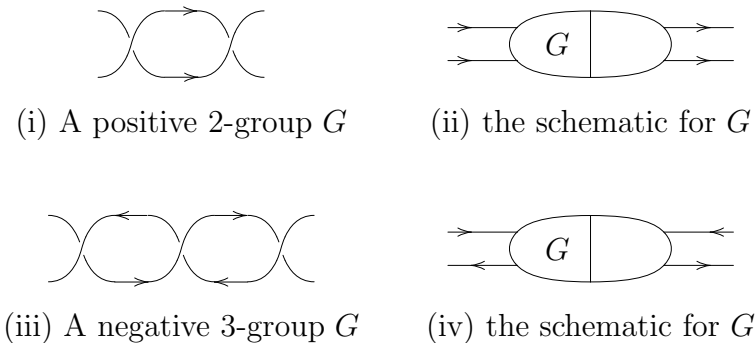


For an integer  $k > 1$ , a component  $k$ -group,  $k > 1$ , shall be referred to as a positive group if, during any traversal of the component containing the group, both group arcs are traversed in the same order, otherwise the group is said to be negative. Note that we require  $k > 1$  since the convention has no meaning for a component 1-group.

If the link is oriented, then we shall refer to link groups as positive or negative according to the same convention (again, this only applies to link groups of size 2 or more).



A 1-group shall be called a loner. We shall see later that, in a knot or an oriented link, there are circumstances which will allow us to declare a loner to be either a negative or a positive group. When a loner is being considered as a positive group, then the two adjacent outbound edges are the two edges at one end of the group, and the two inbound edges are the two edges at the other end of the group, while if the loner is being considered as a negative group, then an inbound edge and the adjacent outbound edge are considered to be the two edges at one end of the group, with the remaining pair of inbound and outbound edges being the two edges at the other end of the group.

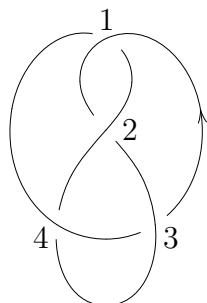


It is clear that each crossing of a link diagram belongs to exactly one group of the link diagram, and so the groups provide us with a partition of the set of crossings of the link diagram.

We are now ready to define the group code for a link diagram.

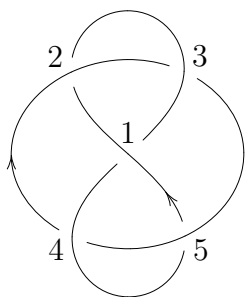
**II.2 DEFINITION.** Let  $D$  denote a diagram of the link  $L$ . For each  $k \geq 1$  for which  $L$  has at least one group of  $k$ , let  $i_k$  be the number of  $k$ -groups. For each such  $k$ , label the  $k$ -groups in an arbitrary fashion with the labels  $k_1, k_2, \dots, k_{i_k}$ . Assign an orientation to each of the  $m$  components of  $L$ . Then arbitrarily choose a component and on that component, arbitrarily choose a group. Select an edge at one end of this group and begin a traversal of the component by following the selected edge into the group. Form the sequence of group labels in the order that the groups are encountered in this traversal. If there are components not yet traversed, place a colon at the end of the sequence as constructed so far and repeat this process. After every component

has been processed, we will have obtained a sequence in which  $m - 1$  colons appear. Each group label will appear exactly twice in this sequence. If a group is negative, then preface both occurrences of the group's label in the sequence with a minus sign. The resulting sequence is called a group code for  $D$ .



Gauss code:  $\overset{+}{1}, \overset{-}{2}, \overset{+}{3}, \overset{-}{4}, \overset{+}{2}, \overset{-}{1}, \overset{+}{4}, \overset{-}{3}$

Group code:  $\underbrace{1, 2}_{-2_1}, \underbrace{3, 4}_{-2_2}, \underbrace{2, 1}_{-2_1}, \underbrace{4, 3}_{-2_2}$



Gauss code:  $\overset{+}{1}, \overset{-}{2}, \overset{+}{3}, \overset{-}{1}, \overset{+}{4}, \overset{-}{5} : \overset{+}{2}, \overset{-}{3}, \overset{+}{5}, \overset{-}{4}$

Group code:  $\underbrace{1, 2}_{1_1}, \underbrace{2, 3}_{2_1}, \underbrace{1, 4}_{1_1}, \underbrace{4, 5}_{-2_2} : \underbrace{2, 3}_{2_1}, \underbrace{5, 4}_{-2_2}$

Diagrams rendered by Knotilus [11]

## CHAPTER III

### FLYPE ORBITS

#### 1. The orbit of a group

Following Menasco and Thistlethwaite [15], we shall, without loss of generality consider all link diagrams to lie on  $S^2$ . Still following Menasco and Thistlethwaite, for each  $n$ -crossing diagram  $D$  of a link  $L$ , we shall take small “crossing-ball” neighbourhoods  $B_1, \dots, B_n$  of the crossing points of  $D$  and, also without loss of generality, we shall assume that  $L$  coincides with  $D$ , except that inside each  $B_i$ , the two arcs of  $D \cap B_i$  are perturbed vertically to form semicircular overcrossing and undercrossing arcs lying on the boundary of  $B_i$ . We shall express this relationship between the link  $L$  and its diagram  $D$  by writing  $L = \lambda(D)$ . A tangle  $T$  of  $L$  is the intersection of a piecewise linear ball  $B$  and  $L$ , where  $B \cap S^2$  is a disc and  $\partial B \cap L$  is finite, nonempty, and lies in  $S^2$ . We shall say that  $T$  is a nontrivial tangle if it contains at least one crossing ball of  $L$ .

Furthermore, we shall say that  $T$  is an  $m$ -tangle if  $|\partial B \cap L| = m$  (note that  $m$  will always be even). Since we shall be largely interested in 4-tangles, from now on, the word tangle with no additional modifier shall mean 4-tangle.

It will be very useful for our work to have a diagrammatic, or graph-theoretic, description of the projection onto  $S^2$  of the tangles of  $L$ . It will suffice for our work to consider those nontrivial tangles whose projection onto  $S^2$  forms a connected subgraph of  $D$ . Such a subgraph of  $D$  will be called a *tangle* of the diagram  $D$ .

Let  $T$  be a nontrivial  $m$ -tangle of  $L$  whose projection onto  $S^2$  is a connected subgraph of  $D$ . Let  $W$  denote the set of crossings in the projection of  $T$  onto  $S^2$ . Then the subgraph of  $D$  induced by  $W$ , denoted by  $D[T]$ , is a connected plane graph in which there exists exactly one face, called the *edge face* of the  $m$ -tangle, for which some number  $k$  of its boundary vertices have degree less than four, with the sum of

the degrees of the boundary vertices equal to  $4k - m$ , while all other vertices have degree 4. Each vertex  $v$  of degree less than four in  $D[T]$  has  $4 - \deg_{D[T]}(v)$  arcs of  $D$  incident to it that lie in the edge face of  $D[T]$ . Such arcs are said to be incident to the tangle.

Conversely, if  $D'$  is a connected induced subgraph of  $D$  such that there exists exactly one face  $E$ , called the *edge face* of  $D'$ , for which some number  $k > 0$  of the boundary vertices of  $E$  have degree less than four while all other vertices of  $D'$  have degree 4, then there exists a nontrivial tangle  $T$  of  $L$  such that  $D' = D[T]$ .

Accordingly, the tangles of the diagram  $D$  are precisely the connected induced subgraphs  $D'$  of  $D$  for which there exists exactly one face  $E$  with some of its boundary vertices having degree less than four while all other vertices of  $D'$  have degree 4.

With this terminology, a link  $L$  is prime if and only if no diagram  $D$  of  $L$  has a 2-tangle; equivalently,  $D$  is 3-edge-connected (that is, the removal of at most two edges does not disconnect the graph).

Thus in a diagram of a prime link, there are no 2-tangles, so  $m = 4$  is the smallest positive integer for which there exists an  $m$ -tangle in a diagram of a prime link.

The following two facts about tangles in a prime link diagram will prove to be quite useful.

**III.1.1 LEMMA.** *Let  $D$  be a diagram of a prime link  $L$ , and let  $T_1$  and  $T_2$  be tangles in  $D$  such that  $T_1 \cap T_2 \neq \emptyset$ ,  $T_1 - T_2 \neq \emptyset$  and  $T_2 - T_1 \neq \emptyset$ . Then there are at least two arcs of  $T_1$  each having exactly one endpoint in  $T_2$ , and at least two arcs of  $T_2$  each having exactly one endpoint in  $T_1$ .*

*Proof.* Since  $T_2$  is connected, there must be at least one arc with one endpoint in  $T_2 - T_1$  and the other endpoint in  $T_2 \cap T_1$ . Suppose there is exactly one such arc. Then the other 3 arcs incident to  $T_1$  must have their other endpoints outside of  $T_2$ . We identify four cases.

Case 1: none of these three arcs have endpoints in  $T_1 \cap T_2$ . Then since  $T_1 \cap T_2$  must have at least 4 incident arcs, there must be at least 3 arcs joining vertices in  $T_1 \cap T_2$  to vertices in  $T_1 - T_2$ , whence three of the four arcs incident to  $T_2$  have already been accounted for. Now there must be at least four arcs incident to  $T_2 - T_1$ , and only one of these can have its other endpoint in  $T_1 \cap T_2$ . Thus there are at least three arcs incident to  $T_2 - T_1$  whose other endpoints are not in  $T_1$ . But then  $T_2$  has at least six incident arcs, which is not possible.

Case 2: exactly one of these three arcs has an endpoint in  $T_1 \cap T_2$ . Then the remaining two arcs are incident to  $T_1 - T_2$ , with their other endpoint not in  $T_2$ . Since  $T_1 - T_2$  must have at least four incident arcs, there must be at least two arcs joining vertices in  $T_1 - T_2$  to vertices in  $T_1 \cap T_2$ . But then we have accounted for at least three arcs incident to  $T_2$ , and all three are actually incident to  $T_1 \cap T_2$ . Since  $T_2 - T_1$  must have at least four incident arcs, of which exactly one is incident to  $T_1 \cap T_2$ , there must be at least three arcs joining vertices in  $T_2 - T_1$  to vertices neither in  $T_2$  nor in  $T_1$ . But then we have accounted for at least six arcs incident to  $T_2$ , which is not possible.

Case 3: exactly two of these three arcs have an endpoint in  $T_1 \cap T_2$ . Then the remaining arc is incident to  $T_1 - T_2$  with its other endpoint not in  $T_2$ . Since  $T_1 - T_2$  must have at least four incident arcs, there must be at least three arcs joining vertices in  $T_1 - T_2$  to vertices in  $T_1 \cap T_2$ . We have now accounted for five arcs incident to  $T_2$ , which is not possible.

Case 4: all three of the arcs are actually incident to  $T_1 \cap T_2$ . Since there are at least four arcs incident to  $T_1 - T_2$ , none of which may have their other endpoints outside of  $T_1$ , there must be at least four arcs joining vertices in  $T_1 - T_2$  to vertices of  $T_1 \cap T_2$ . But then we have accounted for at least seven arcs incident to  $T_2$ , which is not possible.

Since every case leads to a contradiction, we conclude that there must be at least two arcs having one endpoint in  $T_2 - T_1$  and the other endpoint in  $T_2 \cap T_1$ . Since  $T_1$  and  $T_2$  have symmetric roles, it follows as well that there must be at least two arcs having one endpoint in  $T_1 - T_2$  and the other endpoint in  $T_2 \cap T_1$ . ■

**III.1.2** DEFINITION. A tangle whose incident edges belong to the same component is called a component tangle, otherwise it is called a link tangle.

A component tangle may have edges belonging to more than one component which means there are components of the link completely contained within the tangle. A link tangle may also have edges within the tangle belonging to components other than the two components with incident edges to the tangle. These components are contained within the tangle.

**III.1.3** PROPOSITION. *Let  $D$  be a diagram of a prime alternating link  $L$ , and let  $T_1$  and  $T_2$  be tangles in  $D$ . Suppose that there exist arcs  $e$  and  $f$  with endpoints  $a$  and  $b$ , respectively, such that*

- (i)  $a$  and  $b$  are vertices of both  $T_1$  and  $T_2$ ,

(ii)  $e$  and  $f$  are edges incident to both  $T_1$  and  $T_2$ ,

then either  $T_1 \subseteq T_2$  or else  $T_2 \subseteq T_1$ .

*Proof.* Suppose not. Then  $T_1 - T_2$  is an  $m$ -tangle with  $m \geq 4$  since  $L$  is prime. Thus there are at least four arcs incident to  $T_1 - T_2$ . Since  $a, b \in T_2$ , these arcs are all different from  $e$  and  $f$ . But  $T_1$  has four incident arcs, two of which are  $e$  and  $f$ , so at most two of these  $m$  arcs are incident to  $T_1$ , with the having one end point in  $T_1 - T_2$  and the other endpoint in  $T_1 \cap T_2$ . Thus at least two of these  $m$  arcs are incident to  $T_2$ . Since  $T_2$  is a 4-tangle, and  $e$  and  $f$  are incident to  $T_2$ , it must be that there are exactly two arcs that have one endpoint in  $T_1 \cap T_2$ , and the other endpoint in  $T_1 - T_2$ . The same reasoning applied to  $T_2 - T_1$  establishes that there are exactly two arcs that have an endpoint in  $T_1 \cap T_2$  and the other endpoint in  $T_2 - T_1$ . But then the four arcs incident to  $T_1$  are  $e, f$  and the two arcs going from  $T_1 \cap T_2$  to  $T_2 - T_1$ , whence  $T_1 - T_2$  has only two incident arcs. This contradicts the fact that  $m \geq 4$ , so the assumption that neither  $T_1 \subseteq T_2$  nor  $T_2 \subseteq T_1$  holds is false. ■

**III.1.4 COROLLARY.** *If  $e$  and  $f$  are edges in a diagram  $D$  of a prime alternating link  $L$ , and  $a$  and  $b$  are endpoints of  $e$  and  $f$ , respectively, such that there is a tangle  $T$  with  $e$  and  $f$  incident to  $T$  and  $a, b \in T$ , then there is a minimum such tangle.*

*Proof.* Of all tangles with this property, let  $T$  denote one with fewest vertices. By Proposition 3, if  $T_1$  is any tangle with this property, then either  $T \subseteq T_1$  or else  $T_1 \subseteq T$ . If  $T_1 \subseteq T$ , then by the minimality of  $T$ , we have  $T_1 = T$ . Otherwise,  $T \subseteq T_1$ . ■

In [15], Menasco and Thistlethwaite provided a precise definition of flype. Their definition first introduces the notion of a standard flype then uses that to give the general notion of a flype. For convenience we shall provide these definitions here. A diagram  $D$  of the form shown in Figure 1 (a), where the circles illustrated in that figure represent euclidean 2-spheres such that each 2-sphere meets  $S^2$  in a great circle of the 2-sphere, shall be called a standard diagram. Thus the 2-spheres  $S_A$  and  $S_B$  enclose tangles  $A$  and  $B$ , respectively.  $S_x$  is the boundary of a crossing ball and  $S_y$  bounds a ball meeting the link  $\lambda(D)$  in two parallel straight line segments. We remark that neither  $A$  nor  $B$  is required to be nontrivial.

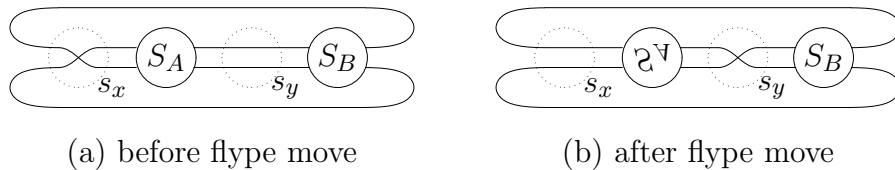


Figure 1.

Let  $D_1$  be a standard diagram. A *standard flype* of  $(S^3, \lambda(D_1))$  is any homeomorphism  $f$  which maps  $(S^3, \lambda(D_1))$  to a pair  $(S^3, \lambda(D_2))$ , where  $D_2$  has the form shown in Figure 1 (b), in such a way that

- (i)  $f$  maps the 3-ball bounded by  $S_A$  into itself through a rigid rotation through  $\pi$  about an axis in the projection plane parallel to the long arcs at the top and bottom of the diagram,
- (ii)  $f$  fixes pointwise the 3-ball bounded by  $S_B$ ,
- (iii)  $f$  maps each of the 3-balls bounded by  $S_x$  and  $S_y$  into itself by a half-twist.

To extend the notion of flype so that it is applicable to all diagrams, we need the notion of a flat homeomorphism. Let

$$N = \{ x \in \mathbb{R}^3 \mid \frac{1}{2} \leq \|x\| \leq \frac{3}{2} \}.$$

$N$  has an obvious product structure,  $N \simeq S^2 \times [1/2, 3/2]$ .

A homeomorphism  $g: (S^3, \lambda(D_1)) \rightarrow (S^3, \lambda(D_2))$  is said to be *flat* if it is pairwise isotopic to a homeomorphism  $h$  such that  $h$  maps  $N$  onto itself and  $h|_N = h_0 \times id_{[1/2, 3/2]}$  for some orientation-preserving homeomorphism  $h_0: S^2 \rightarrow S^2$ .

Let  $D_1$  and  $D_2$  be diagrams. A flype is any homeomorphism  $f: (S^3, \lambda(D_1)) \rightarrow (S^3, \lambda(D_2))$  of the form  $f = g_1 \circ f' \circ g_2$ , where  $f'$  is a standard flype and  $g_1, g_2$  are flat.

The main theorem in [15] is the following:

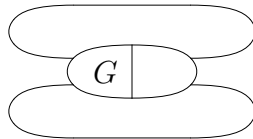
**III.1.5 THEOREM** (Tait Flyping Conjecture). *Let  $D_1$  and  $D_2$  be, reduced, prime, oriented, alternating link diagrams and let there be an orientation-preserving homeomorphism of pairs  $f: (S^3, \lambda(D_1)) \rightarrow (S^3, \lambda(D_2))$ . Then  $f$  is a composite of flypes.*

It is clear that any flype from  $(S^3, \lambda(D_1))$  to  $(S^3, \lambda(D_2))$  corresponds to a traditional diagrammatic flype from  $D_1$  to  $D_2$ . On a historical note, it appears that the modern usage of the word “flype” dates back to J.H. Conway [3], whereas Tait used the term flype to mean a face inversion. Tait used the term “twist” for what we now call a flype.

Our work relies heavily on the fact that two prime alternating links  $\lambda(D_1)$  and  $\lambda(D_2)$  are equivalent (that is, there exists an orientation-preserving homeomorphism from  $(S^3, \lambda(D_1))$  to  $(S^3, \lambda(D_2))$ ) if and only if  $D_1$  and  $D_2$  are (diagrammatic) flype equivalent.

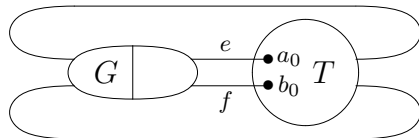
We are now ready to present the definition of the orbit (or flype circuit) of a group in a prime alternating link.

**III.1.6 DEFINITION.** Let  $D$  be a diagram of the prime alternating link  $L$ , and let  $G$  be a group of  $D$ . If  $L$  is a torus link, then  $D$  is

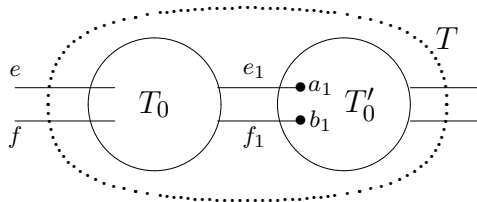


and we define the orbit of  $G$  to be the sequence  $(G)$ .

Otherwise,  $L$  is not a torus link, and there exists a nontrivial tangle  $T$  such that  $D$  has the form

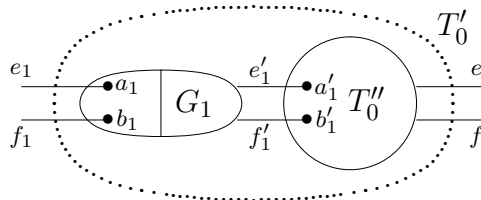


Since  $G$  is a group,  $a_0 \neq b_0$ . Let  $G_0^* = G_0 = G$ . By Corollary 4, there is a tangle  $T_0$  which contains  $a_0, b_0$  and has  $e, f$  as incident edges, and which is contained in all other tangles with this property. In particular,  $T_0 \subseteq T$ . Either  $T_0 = T$  or there exists a nontrivial tangle  $T'_0$  such that  $T$  has the form

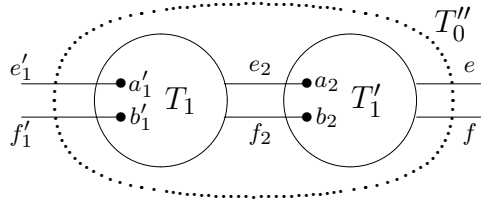


If  $T_0 = T$ , then the orbit of  $G$  is the sequence  $(G_0^*, T_0)$ . Otherwise, we have the nontrivial tangle  $T'_0$  and we examine the arcs  $e_1$  and  $f_1$ .

Case (1):  $a_1 = b_1$ . Let  $G_1$  denote the group containing this crossing. Then  $G_1 \subseteq T'_0$ . Since  $G$  is a group,  $G_1 \neq T'_0$  and so there exists a nontrivial tangle  $T''_0$  such that  $T'_0$  has the form

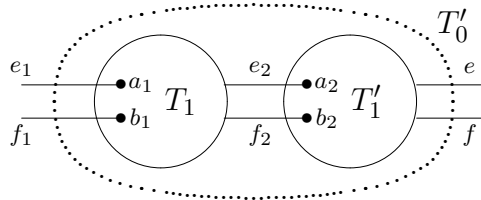


By Corollary 4, there is a tangle  $T_1$  which contains  $a'_1, b'_1$  and has  $e'_1, f'_1$  as incident edges, and which is contained in all other tangles with this property. In particular,  $T_1 \subseteq T''_0$ . Either  $T_1 = T''_0$ , in which case we let  $T'_1$  denote a trivial tangle, or there exists a nontrivial tangle  $T'_1$  such that  $T''_0$  has the form



Let  $\overset{*}{G}_1 = G_1$ .

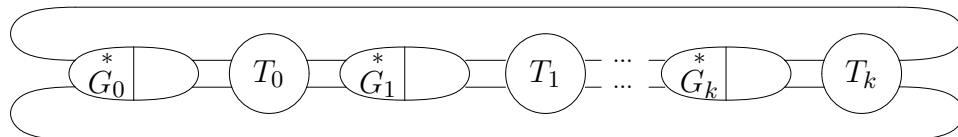
Case (2):  $a_1 \neq b_1$ . By Corollary 4, there is a tangle  $T_1$  which contains  $a_1, b_1$  and has  $e_1, f_1$  as incident edges, and which is contained in all other tangles with this property. In particular,  $T_1 \subseteq T_0'$ . Either  $T_1 = T_0'$ , in which case we let  $T_1'$  denote a trivial tangle, or there exists a nontrivial tangle  $T_1'$  such that  $T_0'$  is of the form



Let  $\overset{*}{G}_1 = \{e_1, f_1\}$ .

In either case, if  $T_1'$  is trivial, we define the sequence  $(\overset{*}{G}_0, T_0, \overset{*}{G}_1, T_1)$  to be the orbit of  $G$ . Otherwise,  $T_1'$  is nontrivial and we repeat the process, replacing  $T_0$  by  $T_1'$ . For some  $k$ , we end up with  $T_k'$  trivial and the process stops. The orbit of  $G$  is the sequence  $(\overset{*}{G}_0, T_0, \overset{*}{G}_1, T_1, \dots, \overset{*}{G}_k, T_k)$ .

The diagram  $D$  has the form



The tangles  $T_0, T_1, \dots, T_k$  are called the *min-tangles* of the orbit of  $G$ , and for each  $i$ ,  $0 \leq i \leq k$ ,  $\overset{*}{G}_i$  marks *position*  $i$  of the orbit. For each  $i$ ,  $\overset{*}{G}_i$  either denotes a group or else a pair of arcs. In either case,  $\overset{*}{G}_i$  is carried on a pair of strands joining two consecutive min-tangles. Each such pair of strands shall be referred to as a *position pair* in the flype orbit of  $G$ . If  $k = 1$ , then  $G$  is said to have a trivial flype orbit, and if  $k > 1$ , then  $G$  is said to be a flying group. If  $G$  is a flying loner, then we shall refer to  $G$  as a negative group (of size 1) if during a link traversal, the two arcs of any position pair are traversed in opposite order, otherwise we refer to  $G$  as a positive group. We remark that if  $G$  is a non-flying loner, then we do not refer to  $G$  as either a positive or negative group.

Let  $G_F = \{G_i^* \mid G_i^* \text{ is a group}\}$ .  $G_F$  is called the *full group determined by  $G$*  and each  $G_i^*$  that is a group is said to be the *subgroup in position  $i$*  of the full group. If  $G_F = \{G\}$ , then  $G_F$  is said to be a *full group*. Otherwise, we say that  $G_F$  is a *split group*. If  $G_F = \{G\}$ , we also say that  $G$  is a full group. When  $G$  is a full group we shall denote the orbit of  $G$  by  $(G, T_0, T_1, \dots, T_k)$ .

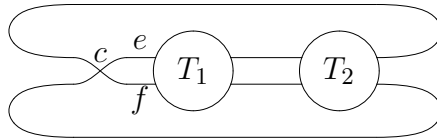
Note that, as a consequence of the definition, each min-tangle in the orbit of a group necessarily contains at least two crossings.

We remark that if the two arcs on the other side of group  $G$  had been chosen to construct the orbit of  $G$ , the sequence that would result would be the present sequence presented in the reverse order. Moreover, if for some  $i$ ,  $G_i^*$  is a group and we used the two arcs from  $G_i^*$  to  $T_i$  to construct the orbit of the group  $G_i^*$ , the result would be the sequence  $(G_i^*, T_i, G_{i+1}^*, T_{i+1}, \dots, G_k^*, T_k, G_0^*, T_0, \dots, G_{i-1}^*, T_{i-1})$ .

**III.1.7 DEFINITION.** A link diagram  $D$  is said to be a *split-group diagram* if there exists at least one group in the diagram that is not full.  $D$  is said to be a *full group diagram* if it is not a split group diagram.

Our next objective is to establish that, in a link diagram  $D$ , if  $G$  and  $G'$  are groups for which  $G_F \neq G'_F$ , then the orbit of  $G$  and the orbit of  $G'$  are in a sense orthogonal to each other. By this we mean that there is a min-tangle  $T'$  of the orbit of  $G'$  such that  $D - T'$  is contained within a min-tangle  $T$  of the orbit of  $G$ . In order to show this, we shall first of all need to identify the possible flype moves for an individual crossing.

**III.1.8 DEFINITION.** Let  $D$  be a diagram of a prime alternating link  $L$ , and let  $c$  be a crossing of  $D$ . A scenario for  $c$  on edges  $e$  and  $f$  is a sequence  $(c, e, f, T_1, T_2)$ , where  $T_1$  and  $T_2$  are nontrivial tangles,  $e$  and  $f$  are adjacent arcs at  $c$ , incident to  $T_1$ , and  $D$  has the form



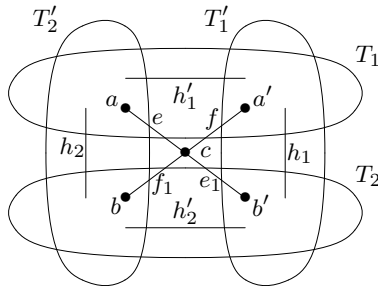
Note that neither  $T_1$  nor  $T_2$  is required to be a min-tangle in the orbit of the group determined by  $c$ , and in particular, it is not implied that the endpoints of  $e$  and  $f$  that belong to  $T_1$  be distinct, nor that the endpoints of the other two arcs incident to  $c$  that belong to  $T_2$  be distinct. For example, if  $L$  is a 3-crossing torus link, then we might take  $c$  to be the first crossing in the group of three crossings,  $T_1$  to be just the

second crossing with its incident arcs and  $T_2$  to be the third crossing with its incident arcs, with  $e$  and  $f$  being the arcs from  $c$  to  $T_1$ .

We show next that if the edges incident to a crossing  $c$  are labelled in the clockwise direction as  $e, f, e_1$  and  $f_1$ , and there is a crossing flype scenario for  $c$  on edges  $e$  and  $f$ , then there is no crossing flype scenario for  $c$  on edges  $f$  and  $e_1$ .

**III.1.9 PROPOSITION.** *Let  $D$  be a diagram of a prime alternating link  $L$  and let  $c$  be a crossing of  $D$  such that the edges incident to  $c$  have been labelled in the clockwise direction as  $e, f, e_1$ , and  $f_1$ . If there is a crossing flype scenario for  $c$  on edges  $e$  and  $f$ , then there is no crossing flype scenario on edges  $f$  and  $e_1$ .*

*Proof.* Suppose to the contrary that  $(c, e, f, T_1, T_2)$  and  $(c, f, e_1, T'_1, T'_2)$  are crossing flype scenarios. Let the endpoints of  $e$  and  $f$  in  $T_1$  be denoted by  $a$  and  $a'$ , respectively, and let the endpoint of  $e_1$  that is in  $T'_1$  be denoted by  $b'$ . Let  $f_1$  denote the fourth edge incident to  $c$ , and denote its other endpoint by  $b$ . Then we have  $a$  and  $a'$  in  $T_1$ ,  $b$  and  $b'$  in  $T_2$ ,  $a'$  and  $b'$  in  $T'_1$  and  $a$  and  $b$  in  $T'_2$ . Let  $h_1$  and  $h_2$  denote the two edges incident to both  $T_1$  and  $T_2$ , and let  $h'_1$  and  $h'_2$  denote the two edges incident to both  $T'_1$  and  $T'_2$ . Since  $a$  and  $a'$  are in  $T_1$ , which is connected, there is a path from  $a$  to  $a'$  using only edges of  $T_1$ . But the graph that is obtained by deleting  $c$  and the two edges  $h'_1$  and  $h'_2$  from  $D$  has two connected components, namely  $T'_1$  and  $T'_2$ , and  $a$  is in  $T'_2$ , while  $a'$  is in  $T'_1$ . Thus every path in  $D$  from  $a$  to  $a'$  that does not go through  $c$  must use either edge  $h'_1$  or else  $h'_2$ .



Since there is a path from  $a$  to  $a'$  that uses only edges in  $T_1$ , either  $h'_1$  or  $h'_2$  is an edge in  $T_1$ . A similar argument applied to  $b$  and  $b'$  shows that either  $h'_1$  or  $h'_2$  must be an edge in  $T_2$ . Thus exactly one of  $h'_1$  and  $h'_2$  belongs to  $T_1$ , the other belongs to  $T_2$ . We may suppose without loss of generality that  $h'_1$  is in  $T_1$  and  $h'_2$  is in  $T_2$ . Similarly, we may suppose without loss of generality that  $h_1$  is in  $T'_1$  and  $h_2$  is in  $T'_2$ . Now  $a$  belongs to both  $T_1$  and  $T'_2$ , so  $T_1 \cap T'_2$  is an  $m$ -tangle for some  $m \geq 4$ . Now  $h'_1$  is an edge of  $T_1$  and incident to  $T'_2$ , so  $h'_1$  is incident to  $T_1 \cap T'_2$ . Similarly,  $h_2$  is incident to  $T_1 \cap T'_2$ . As well,  $e$  is incident to  $T_1 \cap T'_2$ . There must be at least one more edge  $g$  incident to  $T_1 \cap T'_2$ . The edges incident to  $T_1$  are  $e, f, h_1$  and  $h_2$ , and  $g$  is not

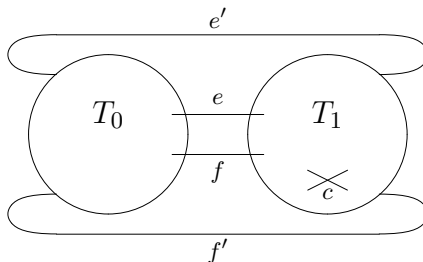
equal to  $e$  or  $h_2$ . Since  $f$  is incident to  $T'_1$  and not to  $T'_2$ ,  $g$  is not equal to  $f$ . Finally,  $h_1$  is an edge in  $T'_1$  and therefore not incident to  $T'_2$ , whence  $g$  is not equal to  $h_1$ . Thus  $g$  must be an edge of  $T_1$ . But now a similar argument shows that  $g$  must also be an edge of  $T'_2$ , whence  $g$  is an edge of  $T_1 \cap T'_2$ . But this contradicts the fact that  $g$  was incident to  $T_1 \cap T'_2$ , and so it is not possible to have crossing flype scenarios  $(c, e, f, T_1, T_2)$  and  $(c, f, e_1, T'_1, T'_2)$ .  $\blacksquare$

This leads to the next result, which may be thought of as establishing the independence of orbits, or the non-interference of orbits.

**III.1.10 THEOREM.** *Let  $D$  be a diagram of a prime alternating link  $L$ , and let  $G$  be a group of  $L$  and  $T_1$  be a min-tangle of the orbit of  $G$ . Let  $c$  be a crossing in  $T_1$  and suppose that  $T$  is a tangle over which  $c$  may flype. Then either  $T \subseteq T_1$  or else  $D - T_1 \subseteq T$ .*

*Proof.* To begin with, we observe that we may assume without loss of generality that  $T \cap T_1 \neq \emptyset$ . To see why this is so, we note that  $c$  may flype over the tangle  $T' = D - T - \{c\}$ , and if  $T \cap T_1 = \emptyset$ , then  $T' \cap T_1 \neq \emptyset$ . Now,  $T \subseteq T_1$  if and only if  $D - T_1 \subseteq D - T$ , and since  $c \in T_1$ , we know that  $c \notin D - T_1$ , whence  $D - T_1 \subseteq D - T$  if and only if  $D - T_1 \subseteq D - T - \{c\}$ . Thus  $T \subseteq T_1$  if and only if  $D - T_1 \subseteq T'$ . Finally, since  $T = D - T' - \{c\}$ , we obtain by symmetry that  $D - T_1 \subseteq T$  if and only if  $T' \subseteq T_1$ . Consequently,  $T \subseteq T_1$  or  $D - T_1 \subseteq T$  if and only if  $D - T_1 \subseteq T'$  or  $T' \subseteq T_1$ .

We have established now that we may assume that  $T \cap T_1 \neq \emptyset$ . If  $T \subseteq T_1$ , then there is nothing to prove, so we consider the case when  $T$  contains at least one crossing that is not in  $T_1$ . We must show that  $D - T_1 \subseteq T$ . Suppose to the contrary that  $D - T_1 \not\subseteq T$ , and let  $T_0 = D - T_1$ . Label one of the position pairs of arcs for  $T_1$  as  $e$  and  $f$ , with the other pair being labelled with  $e'$  and  $f'$ .

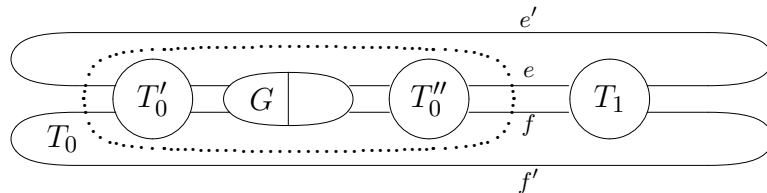


Now,  $T_0 \not\subseteq T$  implies that  $T_0 - T \neq \emptyset$ , whence  $T_0 - T$  is an  $m_1$ -tangle and  $T_0 \cap T$  is an  $m_2$ -tangle for some even integers  $m_1, m_2 \geq 4$ . If there was at most one edge joining a vertex of  $T_0 - T$  to a vertex of  $T_0 \cap T$ , then there would be least 3 edges each with one endpoint in  $T_0 - T$  and the other endpoint not in  $T_0$ . Since  $T_0$  has exactly

4 incident edges, this would mean that there is exactly one edge with one endpoint in  $T_0 \cap T$  and the other endpoint not in  $T_0$ , in which case  $m_2 = 2$ , which is not the case. Thus there are at least two edges each with one endpoint in  $T_0 - T$  and the other endpoint in  $T_0 \cap T$ . Such arcs belong to  $T_0$ , so there are at least two arcs of  $T_0$  that are incident to  $T$ . Similarly, since  $T_1 \cap T \neq \emptyset$  and  $T_1 - T \neq \emptyset$ , there are at least two arcs of  $T_1$  that are incident to  $T$ . But  $T$  is a tangle, so there are exactly four arcs incident to  $T$ , whence there must be exactly two arcs between  $T_0 - T$  and  $T_0 \cap T$ , and exactly two arcs between  $T_1 - T$  and  $T_1 \cap T$ . Thus of the four arcs incident to  $T$ , two are arcs of  $T_0$  and two are arcs of  $T_1$ , whence none of the four arcs  $e, f, e', f'$  are incident to  $T$ . Since  $m_1 \geq 4$ , and there are exactly two arcs from  $T_0 - T$  to  $T_0 \cap T$ , at least two of  $e, f, e', f'$  are incident to  $T_0 \cap T$ , hence have both endpoints in  $T$ . There are two cases to consider: both arcs of one of the position pairs for  $T_1$  belong to  $T$ , or exactly one from each of the two position pairs for  $T_1$  belongs to  $T$ .

Case 1: both arcs of one of the position pairs for  $T_1$  belong to  $T$ . Without loss of generality, we may assume that  $e$  and  $f$  belong to  $T$ . But then  $T_1 \cap T$  is a tangle with  $e$  and  $f$  incident edges, contradicting the minimality of  $T_1$ . Thus this case can't occur.

Case 2: exactly one from each of the two position pairs for  $T_1$  belongs to  $T$ . Suppose that the position pairs have been labelled so that  $e$  and  $e'$  are the two arcs that belong to  $T$ . Since two of the four edges incident to  $T$  are edges of  $T_1$ , and the other two are edge of  $T_0$ , we see that  $f$  and  $f'$  are not in  $T$  nor are they incident to  $T$ . Thus  $e$  and  $f$  have different endpoints in  $T_0$ , and  $e'$  and  $f'$  have different endpoints in  $T_0$ . Thus  $e$  and  $f$  are not edges incident to the group  $G$ , nor are  $e'$  and  $f'$  incident to  $G$ , whence  $T_0$  has a decomposition (obtained from the orbit of  $G$  as a tangle  $T'_0$ , the group  $G$  and a tangle  $T''_0$ , so  $D$  has the structure



Now  $T \cap T''_0$ ,  $T''_0 - T$  and  $T - T''_0$  are all nonempty, so by Lemma 1, there are two arcs of  $T''_0$  incident to  $T$ . Similarly, there are two arcs of  $T'_0$  incident to  $T$ . As well, there are two arcs of  $T_1$  incident to  $T$ , whence we have accounted for six arcs incident to  $T$ . But this is not possible, so the assumption that  $D - T_1 \not\subseteq T$  must be false. We conclude therefore that  $D - T_1 \subseteq T$ . ■

**III.1.11 COROLLARY.** *Let  $D$  be a diagram of a prime alternating link  $L$ , and let  $G$  be a group of  $D$ . If  $T_1$  is any min-tangle from the orbit of  $G$  and  $H$  is any group of  $D$  for which  $H \cap T_1 \neq \emptyset$ , then  $H \subseteq T_1$ .*

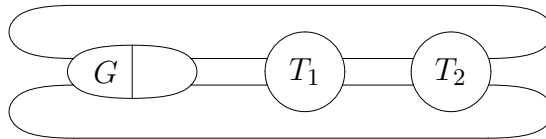
*Proof.* Let  $H$  be a group such that  $H \cap T_1 \neq \emptyset$ , so there is a crossing  $c$  of  $H$  that is in  $T_1$ . Suppose that  $H$  is not contained entirely in  $T_1$ , so there is a crossing  $d$  of  $H$  that is not in  $T_1$ . Let  $T$  denote the connected component of  $H - \{c\}$  that contains  $d$ . Then  $T$  is a tangle over which  $c$  may flype. By Theorem 10, we either have  $T \subseteq T_1$  or else  $D - T_1 \subseteq T$ . Since  $d \in T - T_1$ , it must be that  $D - T_1 \subseteq T$ . But then  $G \subseteq T$ , whence  $d$  is a crossing in  $G$ . By the definition of group, this means that  $c$  is in  $G$ , which contradicts the fact that  $T_1$  is a min-tangle in the orbit of  $G$ . Since this contradiction follows from the assumption that  $H$  is not contained entirely in  $T_1$ , we conclude that  $H \subseteq T_1$ . ■

**III.1.12 COROLLARY.** *Let  $D$  be a diagram of a prime alternating link  $L$ , and let  $G$  be a group of  $D$ . Let  $G'$  be a group in  $D$  other than  $G$  and suppose that  $G$  and  $G'$  are not subgroups of the same split group. Then there exists a min-tangle  $T$  in the orbit of  $G$  such that  $G'$  is contained in  $T$  and each position pair  $P$  of  $G'$  either has both arcs in  $T$  or both arcs are incident to  $T$ , adjacent in a clockwise traversal of the arcs incident to  $T$ , and  $P$  is not a position pair for  $G$ . Moreover, exactly one min-tangle in the orbit of  $G'$  is not contained in  $T$ , and this min-tangle contains  $D - T$ .*

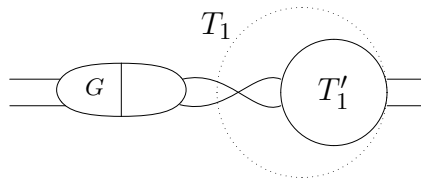
*Proof.* By our choice of  $G'$ , there exists a min-tangle in the orbit of  $G$  such that  $G' \cap T \neq \emptyset$ . By Corollary 11,  $G'$  is contained in  $T$ . Let  $P$  be a position pair for  $G'$ , and suppose that  $P$  is incident to min-tangles  $T_1$  and  $T_2$  in the orbit of  $G'$ . By Theorem 10, either  $T_1 \subseteq T$  or else  $D - T \subseteq T_1$ . Similarly, either  $T_2 \subseteq T$  or else  $D - T \subseteq T_2$ . If both  $D - T \subseteq T_1$  and  $D - T \subseteq T_2$ , then  $D - T \subseteq T_1 \cap T_2 = \emptyset$ , which is not possible. Thus either  $T_1$  and  $T_2$  are both contained in  $T$ , in which case both arcs of  $P$  are arcs of  $T$ , or else exactly one of  $T_1$  or  $T_2$  is not contained in  $T$ . Without loss of generality, suppose that  $T_1$  is not contained in  $T$ , so  $T_2$  is contained in  $T$ . Let the arcs of  $P$  be  $x$  and  $y$ . Then both  $x$  and  $y$  are incident to  $T_2 \subseteq T$  and neither  $x$  nor  $y$  is an arc of  $T$  (since otherwise  $T_1 \cap T$  would be nonempty which would force  $T_1 \subseteq T$ ). Thus  $x$  and  $y$  are incident to  $T$ , adjacent in a clockwise traversal of the edges incident to  $T$ . We claim that  $x$  and  $y$  do not form a position pair in the orbit of  $G$ . Suppose to the contrary that they do. Since  $x$  and  $y$  are incident to  $T_2$ , it follows from Corollary 4 and the definition of the orbit of  $G$  that  $T \subseteq T_2$ . Since  $T_2 \subseteq T$ , this implies that  $T = T_2$ . But  $G'$  is a group in  $T$  that is not in  $T_2$ . This contradiction establishes that  $x$  and  $y$  do not form a position pair in the orbit of  $G$ .

Finally, since  $G$  is contained in some min-tangle in the orbit of  $G'$  there is at least one min-tangle in the orbit of  $G'$  that is not contained in  $T$ . It has been observed above, that there is at most one min-tangle in the orbit of  $G'$  that is not contained in  $T$  and that such a min-tangle in the orbit of  $G'$  must contain  $D - T$ . ■

**III.1.13 DEFINITION.** A *flype scenario*  $(G, T_1, T_2)$  in a link diagram  $D$  consists of a group  $G$  and two nontrivial tangles  $T_1$  and  $T_2$  such that  $D$  is of the form



Observe that  $T_1$  itself cannot be decomposed in the form



In other words, the two arcs that join the group  $G$  to  $T_1$  have distinct endpoints in  $T_1$ . Similarly, the two arcs that join the group  $G$  to  $T_2$  must have distinct endpoints in  $T_2$ .

It is evident that, given a flype scenario  $(G, T_1, T_2)$ , the two arcs joining  $T_1$  and  $T_2$  form a position pair in the orbit of  $G$  and that each crossing of  $G$  can be flyped to these two arcs. The act of flyping all crossings of  $G$  onto the two arcs joining  $T_1$  and  $T_2$  shall be called a *group flype*. Conversely, suppose that  $c$  is a crossing for which there is a flype move: that is, there is a tangle  $T$  across which  $c$  can be flyped. Let  $G$  denote the group that contains  $c$ . There are several possibilities to be investigated.

Case 1:  $G \cap T = \emptyset$ .

Case 1a)  $G \cup T = D$ . In this case, the flype move has the effect of turning the link diagram over.

Case 1b)  $G \cup T \neq D$ . Then  $T_1 = D - (G \cup T)$  is a nontrivial tangle and  $G$ ,  $T$ , and  $T_1$  constitute a flype scenario.

Case 2:  $G \cap T \neq \emptyset$  (by which we mean there is at least one crossing that is common to  $G$  and to  $T$ ). There are two subcases to be addressed.

Case 2a)  $T \subseteq G$ . In this case, the flype move does not change the link diagram.

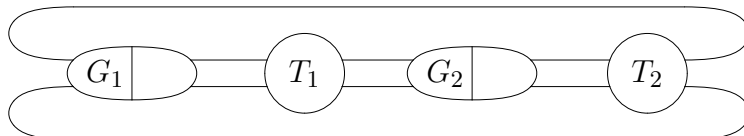
Case 2b)  $T \not\subseteq G$ . Let  $T' = T - G$ , so that  $T'$  is a tangle with  $G \cap T' = \emptyset$ . Furthermore, at least one end of  $G$  is joined to  $T'$ . Let  $c'$  denote the crossing at an

end of  $G$  which is joined to  $T'$ . There is a finite sequence of flype moves, each of which will leave the link diagram unchanged, which will flype  $c$  to the arcs joining  $c'$  to  $T'$ . One final flype move will flype  $c$  across  $T'$ , resulting in the link diagram that is created by flying  $c$  across the original tangle  $T$ . We observe that this is the same diagram that would result from flying  $c'$  across  $T'$ . Accordingly, we replace  $c$  by  $c'$  and  $T$  by  $T'$  to obtain a situation where  $c$  is to be flyped across  $T$  and  $G \cap T = \emptyset$ . This situation has been handled in Case 1.

The preceding discussion makes it clear that the only flype moves of importance are those identified by a flype scenario.

**III.1.14 THEOREM.** *Given a link diagram  $D$  of an alternating prime alternating link  $L$ , there is a finite sequence of group flype moves which will transform  $D$  into a full-group diagram.*

*Proof.* The proof is by induction on the number of groups in the diagram, and the base case would be a link diagram with a single group, which is already a full-group diagram. Suppose now that  $n \geq 1$  is an integer such that for any prime alternating link  $L$ , any diagram of  $L$  with  $n$  groups can be transformed into a full-group diagram of  $L$  by a finite sequence of group flype moves. Let  $D$  be a link diagram with  $n + 1$  groups. If  $D$  is a full-group diagram, then there is nothing to do. Suppose then that  $D$  is a split-group diagram. Let  $G_1$  and  $G_2$  be subgroups of a split group. Then there are nontrivial tangles  $T_1$  and  $T_2$  such that  $D$  has the form



and  $G_2$  can be group flyped over  $T_1$  (or  $T_2$ ) to merge with  $G_1$ . Let  $G'$  denote the group that results when  $G_2$  is group flyped over  $T_1$  to merge with  $G_1$ . Then the orbit of  $G'$  in the resulting link diagram  $D_1(L)$  contains one fewer groups than the orbit of  $G_1$  had in the original diagram  $D$ . If  $H$  is any group of  $D$  that does not belong to the orbit of  $G_1$ , then by Corollary 11,  $H$  is contained within some min-tangle of the orbit of  $G_1$ , whence  $H$  is contained entirely within either  $T_1$  or else  $T_2$ , in which case  $H$  is still a group in  $D_1(L)$ . Thus the diagram  $D_1(L)$  has  $n$  groups. By the induction hypothesis, this diagram can be transformed into a full-group diagram by a finite sequence of group flype moves. Consequently, we obtain a finite sequence of group flype moves that transforms  $D$  into a full-group diagram. Now the result follows by induction. ■

Note that once a link has been given an orientation, it follows that for any  $m$ -tangle, exactly half of the incident arcs are directed into the  $m$ -tangle, with the other half of course directed out of the  $m$ -tangle. In particular, if  $G$  is a group in an oriented prime alternating link diagram  $D$ , then for any min-tangle  $T$  in the orbit of  $G$ , two of the incident arcs are directed into the tangle and two are directed out of the tangle. If  $G$  is a positive group, then both arcs of each position pair for  $T$  are oriented the same, where as we leave the group to traverse the link, the first time that we follow an arc incident to  $T$  establishes that position pair as being directed into  $T$ , while the other position pair will be directed outward from  $T$ . On the other hand, if  $G$  is a negative group, then the two arcs in both position pairs have opposite orientation.

## 2. Flype orbit tangles

**III.2.1 DEFINITION.** Let  $D$  be a diagram for an oriented link  $L$  and  $G$  be a group in  $D$  with orbit  $(G, T_0, \dots, T_k)$  in an oriented diagram  $D$ . A min-tangle  $T_i$ ,  $i \in \{0, 1, \dots, k\}$  is a *core-tangle* of  $G$  if, during a link traversal, both arcs of one position pair incident to  $T_i$  are visited before either arc of the other position pair.

**III.2.2 PROPOSITION.** Let  $D$  be a diagram of a  $L$ , and let  $G$  be a group of  $D$ . The following statements hold.

- (i) If  $G$  is a link group with group arcs on components  $C_1$  and  $C_2$ , then every position pair of  $G$  contains an arc on  $C_1$  and  $C_2$ .
- (ii) If  $G$  is a positive component group (which includes the case of a positive flyping component loner) on component  $C$ , then every position pair of  $G$  has both arcs on  $C$  and any traversal of  $C$  will traverse both arcs of a position pair in the same direction. Moreover, any traversal of  $C$  that starts at one arc of  $G$  will traverse exactly one arc from each position pair before traversing the second arc of  $G$ .
- (iii) If  $G$  is a negative component group (including the case of a negative flyping loner), then both arcs of a given position pair belong to the same component (it's possible that different position pairs may lie on different components) and any traversal of  $C$  will traverse both arcs of a position pair in the opposite direction. Moreover, suppose a position in the orbit of  $G$  is chosen and  $G$  is flyped to that position. Choose any arc  $x$  incident to  $G$  and orient  $C$  so that  $x$  is directed away from  $G$ . In any traversal of  $C$  in the direction of orientation, let  $S$  denote the subwalk from  $G$  leaving on  $x$  up to the next occurrence of  $G$ . Label the min-tangles in the orbit of  $G$  as  $T_0, T_1, \dots, T_k$

so that the orbit of  $G$ ,  $(G, T_0, \dots, T_k)$  is as shown in Figure 1. Then there exists a least  $i \geq 0$  such that neither arc of the position pair from  $T_i$  to  $T_{i+1}$  appears in  $S$  (where  $T_{k+1}$  is taken to be  $T_0$  and the two arcs of the position pair between  $T_k$  and  $T_{k+1} = T_0$  are represented by the two arcs from  $T_k$  to  $G$ ).

- (iv) The orbit of  $G$  contains a core-tangle if and only if  $G$  is a negative component group (which includes the case of a negative flying component loner). Furthermore, if  $G$  is a negative component group (including the case of a negative flying loner), then the number of different components that contain at least one position pair of  $G$  is equal to the number of core-tangles in the orbit of  $G$ .

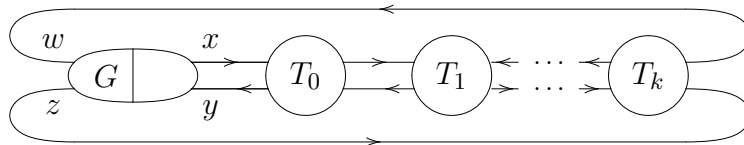


Figure 1. The orbit of a negative component group

*Proof.* Let the orbit of  $G$  be  $(G, T_0, \dots, T_k)$ .

Suppose  $G$  is a link group with arcs on components  $C_1$  and  $C_2$ . Of the two arcs at position 0, one is on  $C_1$  and the other is on  $C_2$ . Suppose now that for some  $i$ ,  $0 \leq i < k$ , of the two arcs at position  $i$ , one lies on  $C_1$  and the other lies on  $C_2$ . Of the four arcs incident to  $T_i$ , two must be from  $C_1$  and two must be from  $C_2$ . Thus of the two arcs at position  $i + 1$ , one must be on  $C_1$  and the other on  $C_2$ . Since the two group arcs in position 0 have this property, the result follows by induction.

Let  $G$  be a positive component group (so  $G$  may be a positive flying component loner) on component  $C$ . Assign a direction of traversal to component  $C$ . Since  $G$  is positive, the two arcs of  $G$  are traversed in the same direction. Suppose now that for some  $i$ ,  $0 \leq i < k$ , both arcs at position  $i$  lie on  $C$  and are traversed in the same direction, either both entering or both leaving  $T_i$ . In either case, the two arcs of the other position pair incident to  $T_i$  must also belong to  $C$  and must be traversed in the same direction, respectively both leaving or both entering  $T_i$ . Since the two group arcs in position 0 have this property, the result follows by induction. Suppose now that during some traversal of  $C$ , in a segment from one arc of  $G$  to the other, there is a position pair neither of whose arcs get traversed. We may assume that the traversal is oriented so that for each  $i$  the arcs of the position pair joining  $T_i$  to  $T_{i+1}$  are oriented from  $T_i$  to  $T_{i+1}$ . Let  $i$  be minimal with respect to the property that neither arc of the position pair from  $T_i$  to  $T_{i+1}$  is traversed during this segment (since

$G$  is incident to  $T_0$ , such  $i$  exists). Due to the minimality of  $i$ , during this segment of the traversal, at least one of the arcs,  $x$  say, entering  $T_i$  is traversed. Since both of the arcs of the position pair containing  $x$  are oriented the same way, we must exit  $T_i$  on arc from the position pair between  $T_i$  and  $T_{i+1}$ . Since we must exit  $T_i$  during this segment of the traversal we have obtained a contradiction. Finally, note that if, during some traversal of  $C$ , there is a position pair both of whose arcs get traversed in a segment from one arc of  $G$  to the other, then in the other segment, neither arcs of this position pair appear. It follows now that for any traversal of  $C$  in a segment from one arc of  $G$  to the other arc of  $G$ , exactly one arc from each position pair will appear.

Let  $G$  be a negative component group (so  $G$  may be a negative flying component loner) on component  $C$ . Assign a direction of traversal to each component of  $D$ . Since  $G$  is negative, the two arcs of  $G$  are traversed in the opposite direction. Suppose now that for some  $i$ ,  $0 \leq i < k$ , both arcs at position  $i$  lie on some component  $C'$  and are traversed in the opposite direction, one entering and one leaving  $T_i$ . Consider the two arcs of the other position pair  $P$  incident to  $T_i$ . Since exactly two arcs must enter  $T_i$  and exactly two must leave  $T_i$ , it follows that of the two arcs of the position pair  $P$ , one ( $x$  say) is entering  $T_i$  and the other ( $y$  say) is leaving  $T_i$ . Let  $C''$  be the component of  $x$ . Since a traversal of  $C''$  starting at  $x$  must leave  $T_i$  by an arc of  $C''$ , it follows that  $y$  lies on component  $C''$  (it is possible that  $C \neq C''$ ).

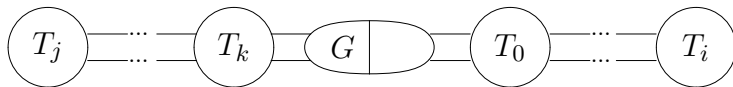
Suppose now that  $G$  has been flyped to some position on component  $C$ , an arc  $x$  incident to  $G$  and an orientation of  $C$  has been chosen so that the orbit of  $G$  is as shown in Figure 1. The subwalk  $S$  is a circuit based at  $G$  whose first arc is  $x$ . The last arc of  $S$  is either  $y$  or  $w$ , since these are the two arcs oriented into  $G$ . If the last arc were  $w$  then  $S$  would be a full traversal of  $C$ , which is not the case. Thus the last arc of  $S$  is  $y$ . By definition of  $S$ , neither  $z$  nor  $w$  appears in  $S$ . For any  $j$ , consider the position pair  $P$  from  $T_j$  to  $T_{j+1}$ . If both arcs of the other position pair incident to  $T_j$  appear in  $S$ , then either both arcs of  $P$  appear in  $S$  or neither arc of  $P$  appears in  $S$ . It follows that there is a smallest  $i$  such that that neither arc of the position pair from  $T_i$  to  $T_{i+1}$  appears in  $S$ .

Suppose now that the orbit of  $G$  contains a core-tangle  $T$ . Assign an orientation to each component in  $L$ . Of the four arcs of the two position pairs incident to  $T$ , two will be oriented into  $T$  and two will be oriented away from  $T$ . Let  $x$  be one which is oriented into  $T$ . Since  $T$  is a core-tangle, a link traversal starting at  $x$  will encounter the other arc  $y$  of the position pair containing  $x$  before encountering either arc of the other position pair. Thus  $y$  is oriented away from  $T$  and so  $x$  and  $y$  have opposite

orientations. It follows that  $G$  is either a negative group or a link group. We claim that  $G$  is not a link group. Suppose to the contrary that  $G$  is a link group. Then, as shown above, there exist components  $C_1$  and  $C_2$  such that for every position pair of  $G$ , one arc of the pair will be on  $C_1$  while the other is on  $C_2$ . Without loss of generality, suppose that  $x$  is on  $C_1$  so that  $y$  is on  $C_2$ . Then a link traversal, starting at  $x$ , must leave  $T$  on  $C_1$  and therefore cannot encounter  $y$  before encountering an arc from the other position pair incident to  $T$ , contradicting the fact that  $T$  is a core-tangle. Thus  $G$  is a negative component group.

Suppose that  $G$  is a negative component group and let  $\{C_1, C_2, \dots, C_r\}$  be the set of components on which  $G$  can flype. Suppose  $t$  is such that  $1 \leq t \leq r$  and  $G$  is in some position on  $C_t$ . Further suppose that the min-tangles in the orbit of  $G$  have been labelled so that the orbit of  $G$  is  $(G, T_0, \dots, T_k)$ . By (iii), there exists a least  $i$  such that on a traversal of  $C_t$  leaving  $G$  in the direction of  $T_0$  but stopping upon the first occurrence of  $G$ , neither arc of the position pair from  $T_i$  to  $T_{i+1}$  is traversed. Then  $T_i$  is a core-tangle for  $G$ , so we have established that the orbit of a negative group has at least one core-tangle.

If we traverse  $C_t$  from  $G$  in the direction towards  $T_k$ , stopping at the first return to  $G$  (we are now thinking of the orbit of  $G$  as  $(G, T_k, T_{k-1}, \dots, T_0)$  and applying (iii) ), then there exists a greatest  $j$  such that neither arc of the position pair from  $T_j$  to  $T_{j-1}$  is traversed. Thus  $T_j$  is a core-tangle and the part of the orbit of  $G$  that contains the position pairs on  $C_t$  is as shown



Since this includes a full traversal of  $C_t$ , there are two possibilities:  $T_i = T_j$  or  $T_i \neq T_j$ . If  $T_i = T_j$ , then  $T_i$  is a component core-tangle and this is the entire orbit of  $G$ . In this case, all position pairs in the orbit of  $G$  lie on  $C_t$  and  $T_i$  is the only core-tangle for  $G$ . Suppose  $T_i \neq T_j$ . Then  $T_i$  and  $T_j$  are link tangles and each has an incident position pair for  $G$  whose arcs do not lie on  $C_t$ . Since all min-tangles that have an incident position pair whose arcs lie on  $C_t$  have been encountered during a full traversal of  $C_t$ .  $T_i$  and  $T_j$  are the only core-tangles of  $G$  that are met during a full traversal of  $C_t$ . Thus, either  $r = 1$  in which case the orbit of  $G$  has a single core-tangle and that core-tangle is a component tangle, or  $r > 1$  and for each  $T$  with  $1 \leq t \leq r$ ,  $C_t$  meets two different core-tangles each of which is a link tangle. In this case, each core-tangle in the orbit of  $G$  is a link tangle meeting two different components from the set  $\{C_1, C_2, \dots, C_r\}$ . Thus the orbit of  $G$  has exactly  $r$  core-tangles.  $\blacksquare$

## CHAPTER IV

### MASTER GROUP CODE

#### 1. Converting a link diagram to a knot diagram

In this section, we introduce the general notion of tangle surgery on diagrams of links and show how we may iteratively use one particular tangle surgery to convert a link diagram to a knot diagram. We remind the reader that by link or knot we mean prime alternating and by diagram we mean minimal; that is, reduced and alternating.

**IV.1.1 DEFINITION.** Let  $D$  be a diagram of a link. Choose any nontrivial tangle  $T_1$  of  $D$ , and replace  $T_1$  by any nontrivial tangle  $T_2$ . Let  $D'$  be the diagram that has been formed. If  $D'$  is alternating, then we say  $D'$  has been formed from  $D$  by a tangle surgery (at  $T_1$ ).

Note that in the formation of  $D'$  in the definition of tangle surgery above, it should be understood that no simplification is to take place.

**IV.1.2 PROPOSITION.** Let  $D$  be a diagram of a link, and let  $T_1$  be a nontrivial tangle of  $D$ . Let  $D'$  be a diagram formed by a tangle surgery on  $D$  at  $T_1$  with replacement tangle  $T_2$ . If  $T_2$  does not contain any 2-tangle, then  $D'$  is prime.

*Proof.* Suppose that  $D'$  is not prime, and let  $T$  be a 2-tangle of  $D$ . If  $T \cap T_2 = \emptyset$ , then  $T$  is a 2-tangle in  $D$ , which is not possible. Thus  $T \cap T_2 \neq \emptyset$ . If  $T \subseteq T_2$ , then we are done, so suppose that  $T - T_2 \neq \emptyset$ . If  $T_2 \subseteq T$ , then we may undo the surgery to form  $T'$  from  $T$  by replacing  $T_2$  with  $T_1$ . But then  $T'$  is a 2-tangle in  $D$ , which is not possible. Thus  $T_2 - T \neq \emptyset$ .

There are at most two arcs incident to  $T \cap T_2$  that are actually incident to  $T$ . There are three cases to consider.

Case 1: there are exactly two arcs incident to  $T \cap T_2$  that are actually incident to  $T$ . Since  $T - T_2$  is an  $m$ -tangle in  $D$ ,  $m \geq 4$ , there are at least four edges incident to

$T - T_2$  each with their other endpoint in  $T \cap T_2$ . But then they account for all four edges incident to  $T_2$ , in which case  $T_2 - T$  has no incident edges. This is not possible.

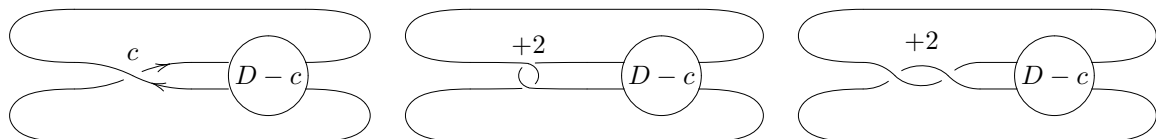
Case 2: there is exactly one arc incident to  $T \cap T_2$  that is incident to  $T$ . Then there is one arc incident to  $T$  that is actually incident to  $T - T_2$ . Since  $T - T_2$  is an  $m$ -tangle in  $D$ ,  $m \geq 4$ , there are at least four edges incident to  $T - T_2$ , exactly one of which has an endpoint outside  $T$ , whence there are at least three edges incident to  $T - T_2$  each with their other endpoint in  $T \cap T_2$ . But this accounts for at least three edges incident to  $T_2$ . Thus there is at most one edge incident to  $T_2$  that has an endpoint in  $T_2 - T$ , which means that  $T_2 - T$  is at most a 2-tangle. But then  $T_2 - T$  must be a 2-tangle, and it is contained in  $T_2$ , as required.

Case 3: there are no arcs incident to  $T \cap T_2$  that are incident to  $T$ . Since  $T - T_2$  is an  $m$ -tangle in  $D$ ,  $m \geq 4$ , and the two arcs incident to  $T$  are actually incident to  $T - T_2$ , there must be at least two arcs from  $T - T_2$  to  $T \cap T_2$ . But this accounts for at least two of the arcs incident to  $T_2$ , whence there are at most two arcs incident to  $T_2$  that are actually incident to  $T_2 - T$ . Any additional arcs incident to  $T_2 - T$  would have to have their endpoints in  $T \cap T_2$ , but these would then be arcs incident to  $T \cap T_2$  and incident to  $T$ , of which there are none. Thus  $T_2 - T$  is a 2-tangle contained in  $T_2$ , as required.

This completes the proof that if  $D'$  is not prime, then  $T_2$  contains a 2-tangle. ■

**IV.1.3 DEFINITION.** Let  $D$  be a diagram of a link  $L$ . A  $\mathcal{D}$  surgery on  $D$  is a tangle surgery where the tangle to be replaced is a group  $G$  in  $D$ , and the replacement tangle is a group of size  $|G| + 1$ . If  $|G| > 1$ , then we require that the end arcs of  $G$  should be the end arcs of the replacement group. The inverse of  $\mathcal{D}$ , denoted by  $\mathcal{D}^-$ , is a tangle surgery where the tangle to be replaced is a non-loner group  $G$  and the replacement tangle is a group of size  $|G| - 1$ . Again, we require that the end arcs of  $G$  should also be the end arcs of the replacement group.

If the original group  $G$  was a loner then the replacement group of size 2 can be inserted in either alignment. We illustrate this situation in Figure 1.



(a) A link loner  $c$  in  $D$

(b) after replacement  
in the positive di-  
rection

(c) after replacement  
in the negative di-  
rection

Figure 1.

We remark that we shall only need to apply the  $\mathcal{D}$  surgery to link groups and correspondingly, we shall only need to apply  $\mathcal{D}^-$  to positive component groups. We illustrate an example of an application of the  $\mathcal{D}$  surgery to a link 2-group in Figure 2.

In Figure 3, we illustrate in general the result of applying  $\mathcal{D}$  to a link group  $G$  on components  $C_1$  and  $C_2$  in a diagram  $D$  of a link  $L$ . Suppose that we have a group code for  $L$ . Without loss of generality, we may suppose that the code for  $C_1$  comes first, then the code for  $C_2$ , and after that, the code for any remaining components.

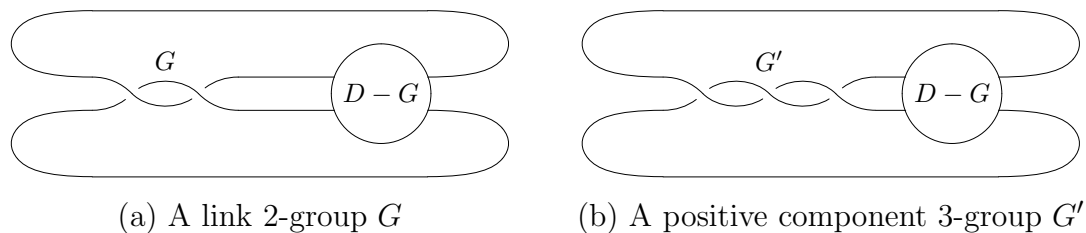


Figure 2.

Furthermore, we may suppose that the codes for  $C_1$  and  $C_2$  both start at  $G$ . Let  $S_1$  and  $S_2$ , respectively, denote the sequences such that the group code is

$$G, S_1 : G, S_2 : \dots$$

Since a group code for a link provides an orientation for each component, and we shall, when working with a group code, always consider each component to have the orientation provided by the group code.

In Figure 3 (a), we have shown just components  $C_1$  and  $C_2$  with a focus on  $G$ . It should be understood that  $D$  may contain other components. Moreover  $C_1$  and  $C_2$  may contain many other crossings so it should be understood that the dotted curves in Figure 3 (a) may very well carry many additional crossings.

Let  $c_1, \dots, c_k$  be the crossings of group  $G$  in Figure 3 (a). In Figure 3 (b),  $G'$  denotes the result of the  $\mathcal{D}$  surgery on  $G$ . We shall treat  $G'$  as if it has been obtained from  $G$  by appending one additional crossing to the group, so that  $G'$  consists of the crossings  $c_1, \dots, c_{k+1}$ . In Figure 3 (a), the first and last edges traversed in  $S_1$  have been labelled  $e$  and  $e'$ , respectively, and the first and last edges traversed in  $S_2$  have been labelled  $f'$  and  $f$ , respectively. Still in (a), a traversal of  $C_1$  that begins with  $e'$  will exit the group on edge  $e$ . Likewise, a traversal of  $C_2$  that begins with  $f$  will enter the group at crossing  $c_k$  and exit the group on the edge  $f'$ .

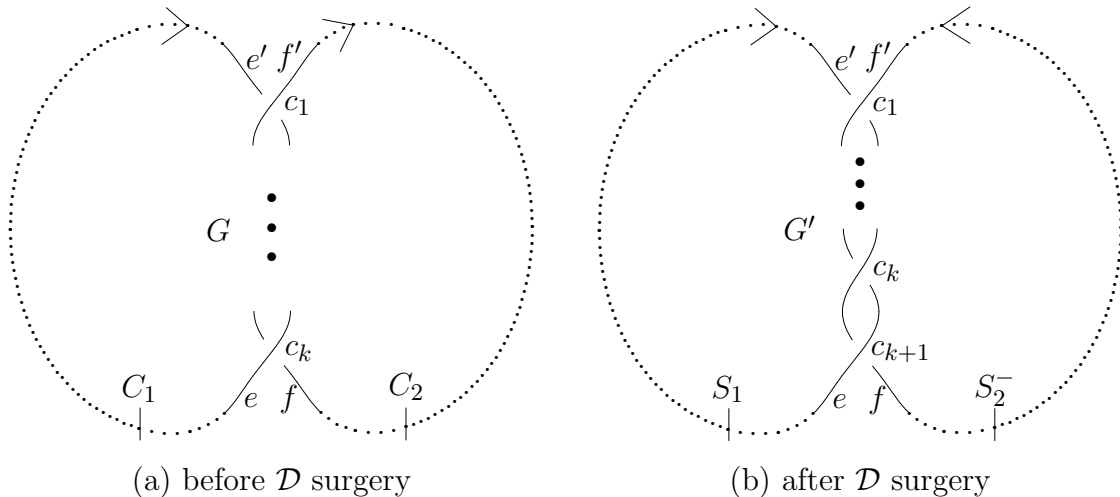


Figure 3. Diagrams rendered by Knotilus.

In Figure 3 (b), begin on edge  $e'$  and traverse into  $G'$ . This time, we exit the group from crossing  $c_{k+1}$  on edge  $f$  (whereas, in (a) we left from crossing  $c_k$  on edge  $e$ ). We have not yet completed a full traversal back to edge  $e'$ . Continuing our traversal will take us through  $S_2$  in the reverse order, returning to the group  $G'$  on edge  $f'$ . We now traverse through  $G'$ , leaving from crossing  $c_{k+1}$  on edge  $e$ . Finally, we traverse  $S_1$  and return to  $e'$ . The result is to merge components  $C_1$  and  $C_2$  into one component and  $G', S_2^-, G', S_1$  is a group code for this component.

Note that we could have begun our traversal in Figure 3 (b) by starting on the edge  $f$  and following the orientation of  $C_2$  to travel into  $G'$ . The resulting group code would be  $G', S_1^-, G', S_2$ . Note further that, this code is simply  $G', S_2^-, G', S_1$  reversed. There is no natural way to choose an orientation for this new component from those of  $C_1$  and  $C_2$ .

We remark that in Figure 3 (a), the orientations of  $C_1$  and  $C_2$  shown made  $G$  into a negative link group. If one of the components been oriented in the reverse direction, then  $G$  would have been a positive link group and just as before, the  $\mathcal{D}$  surgery will cause  $C_1$  and  $C_2$  to be merged into one component with group code  $G', S_2, G', S_1$ .

Whether  $G$  is a positive or negative link group, the  $G'$  that results from an application of the  $\mathcal{D}$  surgery to  $G$  is necessarily a positive component group.

Note that the effect of a  $\mathcal{D}$  surgery to a link group on components  $C_1$  and  $C_2$  may require changes to the group code of any component that meets  $C_1$  or  $C_2$ . Specifically if it was necessary to reverse either  $S_1$  or  $S_2$ , then any link group that had a group

arc on the reversed section and the other arc neither on  $C_1$  or  $C_2$  must have its sign reversed.

**IV.1.4 PROPOSITION.** *Let  $D$  be a diagram for a  $k$ -component link  $L$ , let  $G$  denote a link group of  $D$ , and let  $D'$  denote the diagram of the link  $L'$  that is produced by the application of a  $\mathcal{D}$  surgery to  $G$ . Then  $L'$  has  $k - 1$  components and the following hold.*

- (i) *Let  $g$  be a group of  $D$  other than a subgroup of the full group containing  $G$  and let  $T'$  denote the min-tangle in the orbit of  $g$  that contains  $G$ . Let  $(T')^+$  denote the tangle of  $D'$  that is obtained from  $T'$  by an application of the  $\mathcal{D}$  surgery to  $G$ . Then  $g$  is a group of  $D'$  and  $(T')^+$  is the min-tangle in the orbit of  $g$  in  $D'$  containing  $G'$ . The min-tangles in the orbit of  $g$  in  $D'$  other than  $(T')^+$  coincide with the min-tangles in the orbit of  $g$  in  $D$  other than  $T'$ . Thus the position pairs of  $g$  in  $D$  are the same as the position pairs of  $g$  in  $D'$ .*
- (ii) *If  $G$  is not a loner, or a loner with nontrivial flype orbit such that the two arcs at one end of  $G'$  are both incident to the same min-tangle in the orbit of  $G$  (that is,  $G'$  is aligned in the direction of  $G$ 's orbit), then each subgroup of  $D$ , other than  $G$  itself, of the full group containing  $G$  is a subgroup in  $D'$  of the full group that contains  $G'$ . Furthermore, the min-tangles in the orbit of  $G$  in  $D$  coincide with the min-tangles in the orbit of  $G'$  in  $D'$ , and so the position pairs of  $G$  in  $D$  are the same as the position pairs of  $G'$  in  $D'$ .*
- (iii) *If  $G$  is a loner with trivial flype orbit, then  $G'$  has a trivial flype orbit, independently of the way in which the surgery was performed.*
- (iv) *If  $G$  is a loner with nontrivial flype orbit and, of the two arcs at one end of  $G'$ , one is incident to one min-tangle in the orbit of  $G$  and the other is incident to a different min-tangle in the orbit of  $G$  (that is,  $G'$  is aligned orthogonally to the orbit of  $G$ ), then  $G'$  is a non-flying group of  $D'$ . Furthermore, if  $G$  is a subgroup of a split group, then  $G'$  is a min-tangle in the orbit of each remaining subgroup of the split group. Moreover, each remaining subgroup of the split group is now a negative group (or a loner that flypes in the negative direction) and  $G'$  is a core tangle in its orbit.*

*Proof.* Let  $L$  be a  $k$ -component link and let  $D$  be a diagram for  $L$ . Let  $G$  be a link group in  $D$  and let  $D'$  be the diagram that results from an application of the  $\mathcal{D}$  surgery to  $G$ . It follows from the discussion above that  $L'$  is a  $(k - 1)$ -component link.

For any other group  $g$  of  $D$ ,  $g$  will lie in some min-tangle  $T$  in the orbit of  $G$  and by Corollary III.1.12, there is exactly one min-tangle  $T'$  in the orbit of  $g$  that is not contained in  $T$ . Moreover,  $D - T \subseteq T'$ , and so  $G$  is contained in  $T'$ . In  $D'$ , let  $T''$  be the min-tangle in the orbit of  $g$  which contains  $G'$  and let  $(T')^+$  denote the tangle in  $D'$  that is obtained from  $T'$  by applying the  $\mathcal{D}$  surgery to  $G$ . By Corollary III.1.12,  $D - T \subseteq T''$ . If  $T'' \not\subseteq (T')^+$ , then there exists a crossing  $c$  in  $T''$  that is not in  $(T')^+$ . Since  $D - T \subseteq (T')^+$ , it follows that  $c$  is a crossing in  $T$ . But then, there is a min-tangle  $T_1$  in the orbit of  $g$  in  $D'$  containing  $c$ . Since  $c \notin (T')^+$ , it follows that  $T_1 \subseteq T$  and thus  $T_1 \neq T''$ . However,  $c \in T_1 \cap T''$  and so we have found two distinct min-tangles in the orbit of  $g$  with nonempty intersection, which is not possible. Thus  $T'' \subseteq (T')^+$ . Let  $x$  and  $y$  be arcs of a position pair for  $g$  incident to  $T''$ . The endpoints of  $x$  and  $y$  that are not in  $T''$  are in  $T$  since  $D - T \subseteq T''$ . Since  $T'' \subseteq (T')^+$ , the endpoints of  $x$  and  $y$  in  $T''$  belong to  $(T')^+$ . Thus  $D - T'' = D - (T')^+$  and so  $T'' = (T')^+$ .

The proof of (ii) is immediate (see Figure 4).

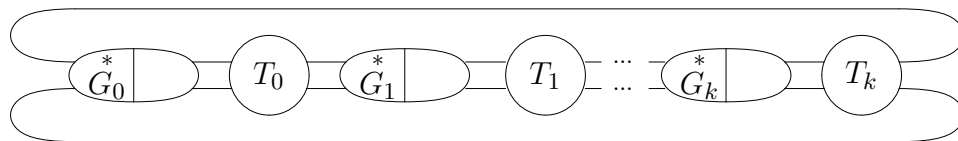


Figure 4.

Suppose that  $G$  is a loner with trivial flype orbit. In  $D'$ , let  $T$  be a min-tangle in the orbit of  $G'$ . Then  $T \subseteq D' - G' = D - G$  and so  $G$  can flype over  $T$  in  $D$ . Thus  $T = D - G = D' - G'$ .

Let  $G$  be a loner with nontrivial flype orbit such that of the two arcs at one end of  $G'$ , one,  $x$  say, is incident to one min-tangle in the orbit of  $G$  and the other,  $y$  say, is incident to a different min-tangle in the orbit of  $G$ . Suppose that,  $G'$  has a nontrivial flype orbit and the two arcs  $x$  and  $y$  form a position pair in the orbit of  $G'$ . Let  $T$  be the min-tangle in the orbit of  $G'$  to which  $x$  and  $y$  are incident. Then  $T \subseteq D' - G' = D - G$  and  $G$  can flype across  $T$ . By Proposition III.1.9, so  $T = D - G = D' - G'$  and thus  $G'$  does not flype. Suppose that  $G$  was a subgroup of a split group and  $G_1$  is another subgroup of the same split group. Then  $G_1$  is a group in  $D'$  and every min-tangle in the orbit of  $G$  is still a min-tangle in the orbit of  $G_1$  in  $D'$ . Let  $T, T'$  be the min-tangles in the orbit of  $G$  to which  $x$ , respectively  $y$ , are incident. Let  $z$ , respectively  $w$ , be the other arc of the position pair in  $G$ 's orbit that contains  $x$ , respectively  $y$  (as shown Figure 5).

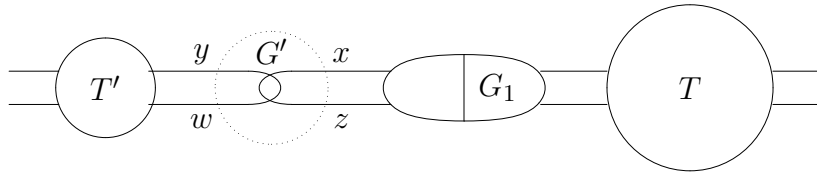


Figure 5.

Then  $x$  and  $z$  constitute a position pair for  $G_1$  and  $y$  and  $w$  constitute a position pair for  $G_1$ . Thus  $G'$  is a tangle over which  $G_1$  can flype and since  $G'$  has two crossings, it is a min-tangle in the orbit of  $G_1$ . Moreover, it is evident from Figure 5 that  $G_1$  is a negative group (or a loner that flypes in the negative direction) and that  $G'$  is a core tangle in the orbit of  $G_1$ . ■

## 2. Master group code

In this section, we introduce the notion of a master group code for a link. As we shall see, a master group code can be constructed from any diagram for the link. A master group code is a cyclic code which contains all flype information for the link, and from it, any diagram of the link can be constructed. We remark that, in general, there are many master group codes for a given link.

We have observed that once the orbits of all groups in a diagram for a link are known, then it is a simple matter to convert the diagram into a full group diagram for the link (see Theorem III.1.14). More generally, once the groups and their orbits are identified, then every diagram of the link can be constructed by flyping the various crossings of the different groups, wherein each crossing can be placed at any position in the orbit of the group to which that crossing belongs. It turns out that there are significant theoretical and computational advantages to working with full group diagrams. The master group code contains within it group codes for all full group diagrams of the link.

For the construction of a master group code, we shall need the following consequence of the non-interference of orbits (see Theorem III.1.10).

**IV.2.1 THEOREM.** *Let  $D$  be a full group diagram of a prime alternating link, and let  $G_1, G_2, G_3$  and  $G_4$  be distinct groups in  $D$  such that  $G_3$  and  $G_4$  are connected by an arc  $e$  and that  $G_1$  can group flype to  $e$  and some other arc, and  $G_2$  can group flype to  $e$  and some other arc. Then  $G_3$  and  $G_4$  are in adjacent min-tangles  $T_1$  and  $T_2$ , respectively, in the orbit of  $G_1$ , and they are in adjacent min-tangles  $S_1$  and  $S_2$ , respectively, in the orbit of  $G_2$ .*

Furthermore, the following hold

- (i)  $G_1$  is contained in either  $S_1$  or  $S_2$ , and  $G_2$  is contained in either  $T_1$  or  $T_2$ .
- (ii)  $G_1$  is in  $S_1$  if and only if  $G_2$  is in  $T_2$ .
- (iii)  $G_1$  is in  $S_1$  if and only if for any sequence of group flypes applied to  $D$  which results in both  $G_1$  and  $G_2$  being group flyped to lie on  $e$ , the link traversal out from the group  $G_3$  in the direction of  $e$  encounters  $G_1$  before  $G_2$ .

*Proof.* Edge  $e$  belongs to a position pair in the orbit of  $G_1$ . Let  $T_1$  and  $T_2$  denote the min-tangles in the flype orbit of  $G_1$  that are joined by this position pair, where the labelling has been chosen so that  $G_3$  is in  $T_1$  and  $G_4$  is in  $T_2$ . Similarly,  $e$  belongs to a position pair in the orbit of  $G_2$ , and we shall let  $S_1$  and  $S_2$  denote the min-tangles in the flype orbit of  $G_2$  that are joined by this position pair, where the labelling has been chosen so that  $G_3$  is in  $S_1$  and  $G_4$  is in  $S_2$ .

$G_2$  belongs to some min-tangle of the orbit of  $G_1$ , and by Theorem III.1.10,  $G_2$  can't flype to any arcs that are not incident to or contained within this min-tangle. Since  $G_2$  can flype onto arc  $e$ , which is incident to the two min-tangles  $T_1$  and  $T_2$  of the orbit of  $G_1$ , it follows that  $G_2$  is contained either in  $T_1$  or else  $T_2$ . Similarly,  $G_1$  belongs either to  $S_1$  or to  $S_2$ .

Suppose that  $G_1$  is in  $S_1$ . We wish to show that  $G_2$  is in  $T_2$ . Suppose to the contrary that  $G_2$  is in  $T_1$ . Any position pair for  $G_2$  is either in or incident to  $T_1$ . Since there is a group flype which will move  $G_2$  onto  $e$ , let us apply it. Let  $e'$  denote the arc connecting  $G_3$  to  $G_2$  and  $e''$  denote the arc connecting  $G_2$  to  $G_4$  (in effect,  $e$  has been replaced by  $e'$  and  $e''$ ). Since  $G_2$  is in  $T_1$ , after flyping it remains in  $T_1$  and so  $e''$  is the arc connecting the min-tangle  $T_1$  (or rather, the tangle flype equivalent to  $T_1$  that was obtained by applying the group flype to  $G_2$ ) to the min-tangle  $T_2$ . Thus we may group flype  $G_1$  to  $e''$  and the other arc joining  $T_1$  to  $T_2$ , so let us perform this flype. The orbits of  $G_1$  and  $G_2$  are not changed by the group flyping of  $G_1$  and  $G_2$ , in the sense that the groups that make up a given min-tangle  $T$  of a given group are still the groups that make up the min-tangle after any group flype whatsoever—they may have changed location, but they have not left the min-tangle  $T$ . In this last diagram, as we traverse from  $G_4$  towards  $G_3$  along the strand that was  $e$ , we encounter first  $G_1$ , then  $G_2$ , and finally  $G_3$ . Thus at one end,  $G_2$  is incident to  $G_1$ , and at the other end,  $G_2$  is incident to  $G_3$ . But  $G_1$  and  $G_3$  are in  $S_1$ , a min-tangle in the orbit of  $G_2$ , which means  $G_2$  has a trivial flype orbit. Since this is not the case, we have obtained a contradiction stemming from the assumption that  $G_2$  was in  $T_1$  and thus  $G_2$  is in

$T_2$ . The symmetry in this argument allows us to conclude similarly that  $G_2$  in  $T_2$  implies that  $G_1$  is in  $S_1$ .

For the last part, we again suppose that  $G_1$  is in  $S_1$ . Apply any sequence of group flypes which results in both  $G_1$  and  $G_2$  being group flyped to the arc  $e$ . Then as we traverse the link from  $G_3$  in the direction of  $e$ , we remain in the min-tangle  $S'_1$  of the orbit of  $G_2$  which is the image of  $S_1$  under this sequence of group flypes. Since  $G_1$  lies on  $e$ , we will encounter  $G_1$  before leaving  $S'_1$ , hence will encounter  $G_1$  before encountering  $G_2$  ( $S'_1$  is a min-tangle in the orbit of  $G_2$ ).

Suppose now, that there is a sequence of group flypes that results in both  $G_1$  and  $G_2$  being flyped onto arc  $e$  such that when we traverse from  $G_3$  along  $e$ ,  $G_1$  is encountered before  $G_2$ . Then  $G_1$  and  $G_4$  lie in different min-tangles of the orbit of  $G_2$ . Since  $G_4$  is in  $S_2$ , this implies that  $G_1$  is in  $S_1$ , as required. ■

The gist of Theorem 1 above is that whenever two or more groups have flype positions with one edge  $e$  in common, and the groups are group flyped to the position in their respective orbits at which  $e$  appears, then the order in which they appear on that strand in any traversal of the link is independent of the order in which the group flyping took place.

**IV.2.2 DEFINITION.** A *master group code* for a prime alternating link  $L$  is a full group code for  $L$  that is augmented with all flype orbit information. It is constructed from a full group diagram  $D$  of  $L$  according to the following procedure. Let  $C$  be any group code for the diagram. Recall that in the construction (see Definition II.0.2 ) of the group code, each group  $G$  was assigned a label of the form  $\pm m_n$ , where  $m$  denotes the size of the group and  $n$  was an index used to distinguish between different groups of the same size. Each pair of consecutive group labels in the group code (recall, this code is cyclic with respect to each component, so the last group label of the code for each component is followed by the first group label of the code for that component) represents the arc that joins these two groups. For each group  $\pm m_n$ , list the pairs of arcs which separate consecutive min-tangles in the orbit of the group (excluding the arcs which connect the group itself to the min-tangles on either side of the group). If the orbit is trivial, there will be only one min-tangle in the orbit and so the list will be empty, in which case the two occurrences of the labels  $m_n$  are replaced by  $m_n^0$ , keeping whatever sign the label had originally. In the case when the orbit is nontrivial, there will be one or more pairs of arcs in the list. In this case, let  $k$  denote the number of pairs in the list. Form the labels  $m_n^i$ ,  $i = 0, \dots, k$ . Choose one of these  $k + 1$  labels and replace both occurrences of  $m_n$  in the group code by the selected

label  $m_n^i$ , keeping the sign of the label being replaced. Then in an arbitrary fashion, assign the remaining  $k$  labels to the  $k$  pairs of arcs in the list. For each arc of a pair, place the label assigned to the pair between the two consecutive groups in the group code which identify the arc. If the group is a negative group, both arcs should receive a minus sign as part of the assigned label. As well, at this point, each loner's flype orbit is known, and those loners with nontrivial flype orbit in the negative direction are to be referred to as negative groups. Such loners shall also receive a minus sign as part of the assigned label. In the event that more than one label is inserted between two consecutive entries of the group code, the labels must be arranged in the uniquely determined order (see Theorem 1) in which the groups would appear if all were group flyped to their respective orbit positions which involved the arc in question.

The sequence of cyclic arrangements of labels for each component that results from this process is called a *master group code* for the link.

In practice, the cyclic arrangement for each component is written as a sequence, with the understanding that the initial entry of the sequence follows the last entry of the sequence.

Since two full group diagrams for the same link only differ in the position in its orbit that each full group is placed, it follows that all full group diagrams for a link can be constructed from the data stored in any master group code for the link (and, of course, all split group diagrams can be so obtained as well).

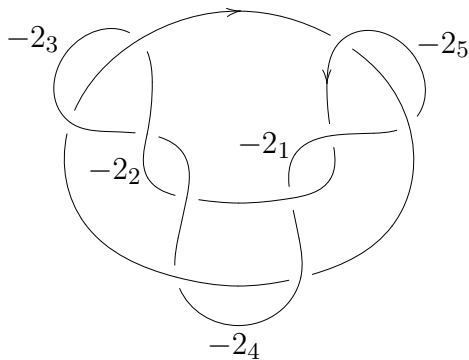
### 3. An algorithm for the construction of a master group code

Given a group code for a diagram  $D$  of a  $k$ -component link  $L$ , we shall apply the following procedure to convert  $D$  into a (minimal) diagram for a (prime alternating) knot.

Let  $D_0 = D$ . Then for each  $i \geq 0$ , if  $D_i$  is not a knot diagram, then form  $D_{i+1}$  by selecting any link group and apply the  $\mathcal{D}$  surgery to it.

It is clear that after exactly  $k-1$  steps, we will have obtained  $D_{k-1}$ , a knot diagram.

For example,  $-2_1, -2_2, -2_3, -2_2, -2_4, -2_1, -2_5 : -2_5, -2_4, -2_3$  is a group code for the two component link shown in Figure 1.



A diagram for the 10 crossing, two component link with Knotilus archive number 10x-2-65, or Thistlethwaite number L10a87.

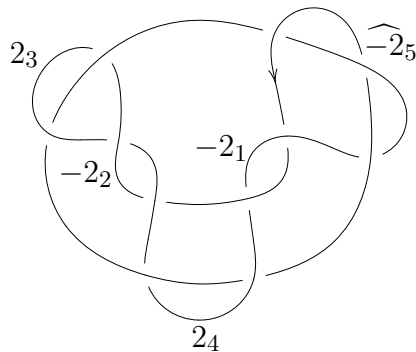
Diagram rendered by Knotilus

Figure 1.

We have decided to apply the surgery  $\mathcal{D}$  to the link group labelled  $-2_5$ , which results in the group code

$$-2_1, -2_2, 2_3, -2_2, 2_4, -2_1, \widehat{-2}_5, 2_3, 2_4, \widehat{-2}_5$$

for the knot shown in Figure 2.



A diagram of a prime alternating 11 crossing knot

Diagram rendered by Knotilus

Figure 2.

We shall denote the group that results from an application of the  $\mathcal{D}$  surgery to a link group  $G$  by  $\widehat{G}$ . Recall that  $\widehat{G}$  is necessarily a positive component group and that the  $\mathcal{D}$  surgery on  $G$  caused the two components of  $G$  to be merged into one. If  $G$  was a negative link group, then the application of the  $\mathcal{D}$  surgery necessitated the reversal of one of the two components, while  $G$  was a positive link group, no reversal was necessary. In the case that  $G$  was a loner, it is possible to apply the  $\mathcal{D}$  surgery in two

different directions, and we shall always choose to apply it in the positive direction; that is, the direction that does not require a component reversal.

By Proposition 1.4, each component group of  $D$  and each link group of  $D$  that is not a subgroup of a split group for which some subgroup had the  $\mathcal{D}$  surgery performed on it will still be a group in  $D' = D_{k-1}$  and its position pairs in  $D$  are the same as its position pairs in  $D'$ . Furthermore, for every nonloner link group  $G$  in  $D$  to which the  $\mathcal{D}$  surgery was applied, or flying loner  $G$  in  $D$  to which the  $\mathcal{D}$  surgery was applied and  $\widehat{G}$  has nontrivial flype orbit in  $D'$ , the position pairs of  $\widehat{G}$  in  $D'$  are exactly the same as the position pairs of  $G$  in  $D$ . Additionally, if  $G$  was a subgroup of a split group in  $D$ , then every other subgroup  $G_1$  of the split group containing  $G$  is still a group in  $D'$  with the same position pairs as those of  $G$  in  $D$  and  $\widehat{G}$  is a subgroup of a split group in  $D'$  whose other subgroups are the other subgroups of  $G$  in  $D$ . Finally, if  $G$  is a loner to which the  $\mathcal{D}$  surgery was applied and  $\widehat{G}$  does not flype and if  $G$  was a subgroup of a split group in  $D$ , then every other subgroup  $G_1$  of that split group in  $D$  is a group in  $D'$  whose position pairs in  $D'$  are the same as its position pairs in  $D$ , except that the two arcs at either end of  $G$  in  $D$  have now become position pairs in the orbit of  $G_1$  in  $D'$  and  $\widehat{G}$  has become a min-tangle in the orbit of  $G_1$  in  $D'$ .

Accordingly, we shall parse the orbit of each group in  $D'$  (by which we mean we shall locate all position pairs of the group). With this information, we construct a master group code for the knot. The final step is to use the information presented above to convert the master group code for the knot into a master group code for  $L$ .

We now describe in detail the algorithm for the construction of a master group code for a knot  $K$ . This algorithm is an adaptation of the algorithm presented in [20], with the major change being that we are aware of groups of the form  $\widehat{G}$ .

We start with an array that initially consists of a group code for a diagram of the knot. In a sense, the group code serves as a skeleton for the master group code that is to be constructed. The first step is to make a list of all the groups in the knot diagram. Then for each group in the list, we identify the min-tangles of the orbit of the group. It will be advantageous to leave the parsing of the flype orbits of groups  $\widehat{G}$  that are of size two and all loners until all other groups have been processed (processing loners before processing groups of the type  $\widehat{G}$ ), because it could be the case that these groups were subgroups of a split group respectively, where another subgroup  $G$  of size two or more discovers this situation during the parsing of their respective flype orbits.

For each group that is a subgroup of a split group, the various subgroups will be group flyped as they are encountered during the orbit identification process to the position of the first encountered subgroup. Each subgroup of the split group other than the first one will be removed from the list of groups whose orbits are to be determined. In the case that one of the subgroups is a  $\widehat{G}$  for some  $G$ , then the resulting full group label will be given a hat in order to retain the information that the  $\mathcal{D}$  surgery was performed on this group (we remark that the size field of the group will be calculated using the original size of  $G$ ; that is, the additional crossing that was created by the application of  $\mathcal{D}$  is not explicitly counted, but rather its presence is indicated by the hat notation). When the parsing of the orbit is completed, the group will be full.

Additionally, if during the parsing of the orbit of  $G'$  where  $G'$  is a negative group or a loner being parsed in the negative direction, we discover that the core tangle is a single group, then we examine this group before we actually start to parse for the min-tangles. If the group is either not of the form  $\widehat{G}$  or is of the form  $\widehat{G}$  for some  $G$  of size greater than one, then this group has trivial flype orbit and so we remove this group from the list of groups to be processed. On the other hand, if the group is of the form  $\widehat{G}$  for  $G$  a loner, then we realize that we should have applied  $\mathcal{D}$  to  $G$  in the direction orthogonal to the present direction. Accordingly, we do not yet parse for the orbit of  $G'$  but instead, we apply  $\mathcal{D}^-$  to  $\widehat{G}$  (resulting in a 2-component link), then apply  $\mathcal{D}$  to  $G$  in the orthogonal direction (resulting in a knot), naming the resulting group  $\widehat{G}$  again. We remark that this step is actually accomplished by simply reversing one of the two sections of code lying between the two arcs of  $\widehat{G}$ . We are now ready to parse for the orbit of  $G'$ , which is now a positive group ( $G'$  is now a subgroup of a split group with  $\widehat{G}$  being another subgroup of the split group, and this will be rediscovered during the parsing of the orbit of  $G'$ ).

After all the groups other than those of the form  $\widehat{G}$ ,  $G$  a loner, have been processed, we begin the processing of these remaining groups (that is, those that have not been removed by the process above). For each remaining group  $\widehat{G}$ , we parse for its orbit. If the orbit turns out to be trivial, then we apply the  $\mathcal{D}^-$  surgery to  $\widehat{G}$ , then apply the  $\mathcal{D}$  surgery to the resulting link group of size one (so the net result is a positive component group  $\widehat{G}$  of size 2 oriented in the direction orthogonal to the direction of the original  $\widehat{G}$ ), and then parse for the flype orbit of  $\widehat{G}$ .

At the end of the orbit identification for each group, the size  $m$  of its full group will have been determined. If it is the  $n^{\text{th}}$  group of size  $m$  that has been processed so far, and there were  $k \geq 1$  positions in the orbit of the group, then in an arbitrary

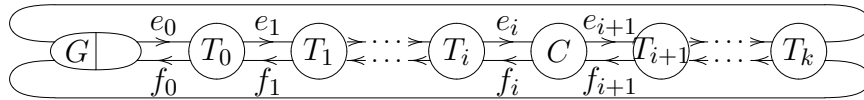
manner, we assign the labels  $\pm m_n^0, \pm m_n^1, \dots, \pm m_n^{k-1}$  to the  $k$  positions (or  $\pm \hat{m}_n^0, \pm \hat{m}_n^1, \dots, \pm \hat{m}_n^{k-1}$  if the group is from the set  $\mathcal{G}(G)$ ). In the array as constructed so far, this is recorded as follows. If the two arcs that separate the two min-tangles which identify the position to which we wish to assign the label  $m_n^i$  are denoted by  $e$  and  $f$ , then we place one copy of the label  $\pm m_n^i$  between the two groups that are the endpoints of  $e$  and one copy of the same label between the two groups that represent the endpoints of  $f$ . From this point on, in the identification of the orbits of other groups still to be analyzed, these two labels of the group's position are treated as if the group were actually residing at that position (this will establish the correct order of flying position on each arc as established in Theorem 2.1). In actual fact, one or both of the "endpoints" of  $e$  and  $f$  may be labels that were deposited during the identification of the orbit of a group that came earlier in the list. As we remarked above, once a label is put in place, then for subsequent group orbit analyses, the label is treated as if it were a group (whose name is the full label, so it will not be confused with any other position label for the same group).

The actual procedure to be followed in the identification of the min-tangles of the orbit depends on whether the group is a positive component group, a negative component group, a loner. If a group is a loner, it is first treated as if it were a negative group and checked to see if it has any flype moves along this alignment. If so, then it does have an orbit along this alignment (and therefore not along its positive alignment), and processing continues to identify the min-tangles of its orbit. On the other hand, if the loner has no flype moves along its negative alignment, it is then treated as a positive group. If it has any flype moves along this alignment, then processing continues to identify the min-tangles of its orbit. If it has no flype moves along either the negative or the positive alignment, then it has a trivial orbit.

The decision to check a loner along its negative alignment first (of course, from a theoretical point of view, the order of checking is not relevant) was empirically based. Our experience has shown the vast majority of loners that do have a nontrivial orbit do so along the negative alignment.

**Case 1: parsing the orbit of a negative group  $G$  or a loner in the negative direction.** In the figure below, we have illustrated the general situation. The core tangle of  $G$ 's orbit is denoted by  $C$  in the figure. The arcs between two consecutive min-tangles have been labelled so that  $e_j$  is the arc going from left to right and  $f_j$  is the arc going the opposite direction. For the sake of clarity, we have drawn the diagram as if the arc  $e_j$  was above the arc  $f_j$ , but of course it could be just the

opposite. It also could be that the orbit has only one min-tangle, namely  $C$ . Of course, this will come out in the analysis of the orbit.



Orbit of a negative component group

If the group code is followed from the first arc of  $G$  to the second arc of  $G$ , then we will traverse both arcs of every group belonging to any min-tangle of the orbit of  $G$  that lies between  $G$  and the core of the orbit of  $G$ ; namely the tangles  $T_0, T_1, \dots, T_i$  as shown in the figure above. Any group for which exactly one of its group arcs is encountered during the traversal from the first arc of  $G$  through the core and back to the second arc of  $G$  must belong to the core itself. If we were to continue the traversal from the second arc of  $G$  through the core and back to the first arc of  $G$  (thereby completing the traversal), all the while making note of the groups for which exactly one of the group arcs was encountered in this part of the traversal, exactly the same collection of groups would be formed.

For simplicity, we shall cycle the current array to bring one of the arcs of  $G$  to the front, and we shall refer to this arc as the first arc of  $G$ . Further, let us refer to the portion of the array that lies between the first and the second arcs of  $G$  as the first section, and the portion which lies between the second arc of  $G$  and the end of the array (the group arc which precedes the first arc of  $G$ ) as the second section.

With this terminology, the first step in the identification of the orbit of  $G$  is to determine which groups of the diagram have one arc in each section. Having done this, we split the analysis into two parts.

**Part I.** This part describes the processing of the second section. It begins with the identification of the two arcs which separate the core from the first min-tangle which follows it ( $e_{i+1}$  and  $f_{i+1}$  the figure above). Observe that this is not the min-tangle that is encountered after the core has been entered for the first time. The traversal that the group code provides starts at the first arc of  $G$  and proceeds through possibly several min-tangles ( $T_0, T_1, \dots, T_i$ ) before entering the core  $C$ . After exiting  $C$  for the first time, we return to the second arc of  $G$  through these same min-tangles (in the reverse order), then through possibly more min-tangles ( $T_k, T_{k-1}, \dots, T_{i+1}$ ) until the core is re-entered by means of arc  $f_{i+1}$ , traversed and exited by means of arc  $e_{i+1}$ , then back through  $T_{i+1}, T_{i+2}, \dots, T_k$  and finally to the first arc of  $G$  to complete the traversal.

Once  $e_{i+1}$  and  $f_{i+1}$  have been identified, the next step will be to identify the entry and exit pair which connect  $T_{i+1}$  to  $T_{i+2}$ , then the pair which connect  $T_{i+2}$  to  $T_{i+3}$  and so on, until we have worked our way completely through the second section. The last pair of arcs that this process identifies is the pair that connects the last min-tangle of this portion of the orbit to the group  $G$ .

So we must begin by finding the two arcs  $e_{i+1}$ ,  $f_{i+1}$  which separate the core from the min-tangle that follows the core in the second section. We have already identified the groups for which one arc lies in the first section and one lies in the second section—these groups we know to be in the core, and we shall refer to them as the *starter core group arcs*. Imagine the strand of the knot which starts at  $f_{i+1}$ , runs through the core, and finishes at  $e_{i+1}$ . This strand will at intervals intertwine with the strand which starts at  $e_i$ , runs through the core and stops at  $f_i$ , thereby forming the starter core groups. As well, it may intertwine with itself to form additional groups. Thus we see that we are searching for the shortest strand which contains the starter group arcs and which is closed in the sense that any non-starter group which has an arc on this strand has both arcs on the strand. To begin with, we extract from the current array the shortest subsequence  $S_1$  that contains all of the starter group arcs. There are two possibilities: either  $S_1$  is closed, or else there exists at least one group which has exactly one of its arcs in  $S_1$ . In the latter case, we extend  $S_1$  to  $S_2$ , the shortest subsequence of the current array that contains the starter arcs and both arcs of each group in  $S_1$  other than the starter groups. If  $S_2$  is not closed, repeat this process. Eventually we arrive at a closed subsequence  $S$  of the current array. Then  $f_{i+1}$  is the arc which leads into the first group arc of  $S$ , and  $e_{i+1}$  is the arc which leads out of the last group of  $S$ .

Next, we must find the pair of arcs  $e_{i+2}$  and  $f_{i+2}$  that separate  $T_{i+1}$  from  $T_{i+2}$  (if indeed there is another min-tangle in this segment of the orbit). Form a new set  $S_1$  by adjoining to the set  $S$  either the group arc that is adjacent to  $e_{i+1}$  but not in the core (that is, not in  $S$ ) or the group arc that is adjacent to  $f_{i+1}$  but not in the core. Again, the selected group arc might be a position marker rather than one of the group arcs from the original group code. Close  $S_1$  as before to obtain a subsequence of the current array. The arcs at each end are the sought-after pair  $e_{i+2}$ ,  $f_{i+2}$ , unless the tangle that lies between the pair  $e_{i+1}$  and  $f_{i+1}$  and the pair just determined is actually another subgroup of  $G$  (it is readily verified that such a tangle is actually a subgroup of the negative group if and only if the tangle itself is just a negative group). In the event that the tangle is a subgroup of  $G$ , we delete this group from the

current array and from the control list, and the number of crossings it contributes to the full group is recorded.

Continue this process until the subsequence that is formed is the entire second section, at which point this part is complete.

**Part II.** This part describes the processing of the first section, whereby the min-tangles which precede the core are identified. This is accomplished by applying the procedure of Part I to the group code (or more precisely, the group code as modified by the construction procedure so far) that is obtained by writing the group code in the reverse order. That is, we construct a temporary working copy of the current state of the master group code by starting at the initial group code, and write the array in reverse order, so that what was called the first section is now the second section. Now apply Part I to this array, but place all labels in the appropriate position in the current array, not the temporary working copy.

We illustrate this process, continuing with the preceding example. We had obtained the prime alternating 11 crossing knot in full group (Knotilus archive number 11x-1-230) shown in Figure 2, with group code

$$-2_1, -2_2, 2_3, -2_2, 2_4, -2_1, \widehat{-2}_5, 2_3, 2_4, \widehat{-2}_5$$

The control list of groups is  $-2_1, -2_2, 2_3, 2_4, \widehat{-2}_5$ , and so we would begin by finding the orbit of the negative group  $G = -2_1$ . No cycling of the array is necessary, since one of the group arcs of  $-2_1$  is the first entry in the array. The first and second sections, respectively, are

$$-2_2, 2_3, -2_2, 2_4 \quad \text{and} \quad \widehat{-2}_5, 2_3, 2_4, \widehat{-2}_5$$

and so the starter group arcs are  $2_3$  and  $2_4$ . Since the sequence  $S_1 = 2_3, 2_4$  is closed,  $S = S_1$  and we have found the entry and exit arcs for this side of the core of the orbit of  $G$  to be the arc from 11 to 5 and the arc from 8 to 9, respectively. In this particular example, the two adjacent arcs to the subsequence  $S$  belong to the same group,  $\widehat{-2}_5$ . We form  $S_1 = \widehat{-2}_5, 2_3, 2_4$ . This is not closed, since the second arc of group  $\widehat{-2}_5$  is missing. Upon closing it up, we obtain  $S = \widehat{-2}_5, 2_3, 2_4, \widehat{-2}_5$ . The tangle which has just been determined has incident arcs  $(5, 11)$  and  $(8, 9)$  as determined earlier, and the arcs  $(1, 9)$  and  $(1, 11)$  which have just been determined. This tangle consists of just the group  $\widehat{-2}_5$ , which must therefore be checked to see if it is a subgroup of the full group determined by  $-2_1$ . Since it is a positive group, it is not a subgroup of

the negative group to which  $-2_1$  belongs, so it is a min-tangle in the orbit of  $-2_1$ . Finally, since  $S$  is equal to the second section, part I has been completed.

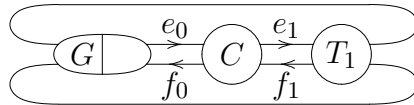
For part II, we reverse the array to obtain

$$-2_1, \widehat{-2}_5, 2_4, 2_3, \widehat{-2}_5, -2_1, 2_4, -2_2, 2_3, -2_2$$

with first and second sections

$$\widehat{-2}_5, 2_4, 2_3, \widehat{-2}_5 \quad \text{and} \quad 2_4, -2_2, 2_3, -2_2,$$

respectively. We know that the starter arcs are  $2_3$  and  $2_4$ , so  $S_1 = 2_4, -2_2, 2_3$ . This is not closed, since the other arc for group  $-2_2$  is missing. Upon closing  $S_1$ , we obtain  $S = 2_4, -2_2, 2_3, -2_2$ , which is the second section. Thus the entry and exit arcs for this side of the core of  $G = -2_1$  are  $(2, 3)$  and  $(8, 2)$ , respectively. Since we have exhausted the second section (the first section of the actual array, since we are doing part II), there are no additional min-tangles. Thus the orbit of  $G = -2_1$  is of the form



orbit of the negative group  $-2_1$

and the updated array, showing the additional position of group  $-2_1$  is

$$-2_1^0, -2_2, 2_3, -2_2, 2_4, -2_1^0, \widehat{-2}_5, -2_1^1, 2_3, 2_4, -2_1^1, \widehat{-2}_5.$$

Now that the orbit of  $-2_1$  has been identified, we can remove it from the control list. The revised control list is  $-2_2, 2_3, 2_4, \widehat{-2}_5$ , and the next group in the list is also a negative group. We cycle the current array so as to start with one of the group arcs for  $G = -2_2$ , say

$$-2_2, 2_3, -2_2, 2_4, -2_1^0, \widehat{-2}_5, -2_1^1, 2_3, 2_4, -2_1^1, \widehat{-2}_5, -2_1^0.$$

The first and second sections are  $2_3$  and  $2_4, -2_1^0, \widehat{-2}_5, -2_1^1, 2_3, 2_4, -2_1^1, \widehat{-2}_5, -2_1^0$ , respectively, and there is only one starter group arc; namely  $2_3$ . We have  $S_1 = 2_3 = S$ , and the entry and exit arcs from this side of the core are  $(11, 5)$  and  $(6, 7)$ , respectively. The adjacent group arcs are  $-2_1^1$  and  $2_4$ , so we choose one, say  $-2_1^1$ , and form  $S_1 = -2_1^1, 2_3$ . Since the other arc of group  $-2_1^1$  is missing,  $S_1$  is not closed. We find the other arc of group  $-2_1^1$  and form  $S_2 = -2_1^1, 2_3, 2_4, -2_1^1$ . Since the other

arc of group  $2_4$  is missing, we find it and form  $S_3 = 2_4, -2_1^0, \widehat{-2_5}, -2_1^1, 2_3, 2_4, -2_1^1$ . Since  $S_3$  is missing the other arc from each of groups  $-2_1^0$  and  $\widehat{-2_5}$ , we form  $S_4 = 2_4, -2_1^0, \widehat{-2_5}, -2_1^1, 2_3, 2_4, -2_1^1, \widehat{-2_5}, -2_1^0$ , which is closed. Thus  $S = S_4$  is the second section, so this part is done. For part II, we reverse the sequence to obtain

$$-2_2, -2_1^0, \widehat{-2_5}, -2_1^1, 2_4, 2_3, -2_1^1, \widehat{-2_5}, -2_1^0, 2_4, -2_2, 2_3,$$

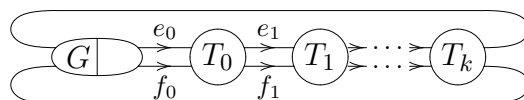
whose second section is simply  $2_3$ . Thus there is nothing to do in part II. The core of the orbit of group  $-2_2$  is just the group  $2_3$ . The updated array, showing the additional position of group  $-2_2$  is

$$-2_2^0, 2_3, -2_2^0, 2_4, -2_1^0, \widehat{-2_5}, -2_1^1, -2_2^1, 2_3, -2_2^1, 2_4, -2_1^1, \widehat{-2_5}, -2_1^0.$$

We delete the group  $-2_2$  from the control list, thereby obtaining the revised control list  $2_3, 2_4, \widehat{-2_5}$ . Since the remaining groups are positive, we must wait until after the method for identifying the orbit of a positive group has been discussed before completing the construction of the master group code for this example.

If we are parsing for the orbit of a loner in the negative direction and we have found a nontrivial flype orbit in this direction, we record this fact by converting the loner into a negative group; for example, if we are processing loner  $1_i$  in the negative direction and a nontrivial orbit has been found, we replace every occurrence of the label  $1_i$  with the label  $-1_i$ .

**Case 2: parsing the orbit of a positive group  $G$  or a loner in the positive direction.** The situation for a positive group is considerably simpler than that of a negative group. As before, we cycle the current array so as to begin with one of the arcs of  $G$ , which we shall refer to as the first arc of  $G$ , and we refer to the portion of the array that lies between the first and the second arcs of  $G$  as the first section, and the portion which lies between the second arc of  $G$  and the end of the array as the second section. The arcs labelled  $e_i$  join group arcs that are in the first section, while the arcs labelled  $f_i$  join group arcs that are in the second section (again, we have taken some artistic liberties, and listed  $e_i$  as if it was always above  $f_i$  in the diagram, but of course this is not necessarily the case).



Orbit of a positive component group

Suppose that we have identified the arcs  $e_i$  and  $f_i$  that identify the  $i^{\text{th}}$  position of the group  $G$  (that is, the two arcs that separate min-tangle  $T_i$  from min-tangle  $T_{i+1}$ , if  $i > 0$ , or the pair of arcs  $e_0$  and  $f_0$  that connect  $G$  to  $T_0$ ). Let  $S_i^1$  denote the sequence of all group arcs that belong to the min-tangles between  $G$  and the position determined by  $e_i$  and  $f_i$  in the order that they are encountered on the strand which leaves  $G$  along  $e_0$ , and let  $S_i^2$  denote the sequence of all group arcs that belong to the same min-tangles but are encountered on the strand which leaves  $G$  along  $f_0$ . Thus  $S_i^1$  is an initial segment of the first section, and  $S_i^2$  is an initial segment of the second section, and these two segments constitute a closed pair, by which we mean that any group that has at least one of its arcs appear in either  $S_i^1$  or  $S_i^2$  also has the other arc appearing in either  $S_i^1$  or  $S_i^2$ . Note that  $S_0^1$  and  $S_0^2$  are empty. If either of  $S_i^1$  or  $S_i^2$  is equal to the section to which it belongs, the orbit has been completely identified. Let  $S_{i+1}^1$  and  $S_{i+1}^2$  be the shortest initial segments of the first and second sections, respectively, for which  $S_i^1$  is a proper initial segment of  $S_{i+1}^1$  and  $S_i^2$  is a proper initial segment of  $S_{i+1}^2$  which are a closed pair in the sense described above. The groups whose arcs appear in  $S_{i+1}^1$  or  $S_{i+1}^2$  but not in  $S_i^1$  or  $S_i^2$  form a tangle which is either a min-tangle in the orbit of  $G$  or a subgroup of the full group determined by  $G$ . It is easy to verify that the tangle is a subgroup of  $G$  if and only if the tangle consists of a single positive group, in which case its two group arcs are deleted from the current array and from the first and second sections, it is removed from the control list, the number of crossings it contributes to the size of the full group to which  $G$  belongs is recorded, and we reset  $S_{i+1}^1 = S_i^1$  and  $S_{i+1}^2 = S_i^2$ , and repeat the process (in effect, we have group flyped the subgroup to amalgamate it with the original group  $G$ ). On the other hand, if the tangle is a min-tangle in the orbit of  $G$ , then we record the pair of arcs  $e_i$  and  $f_i$  as a position pair for the orbit of  $G$ , and set  $e_{i+1}$  to be the arc joining the last group arc of  $S_{i+1}^1$  to the next group arc to be encountered on that strand (either the next group arc in the first section, or, if the first section has been exhausted, to the arc of  $G$  that follows the first section), and set  $f_{i+1}$  to be the arc joining the last group arc of  $S_{i+1}^2$  to the next group arc to be encountered on that strand (either the next group arc in the second section, or, if the second section has been exhausted, to the arc of  $G$  that follows the second section), and repeat the process, replacing  $i$  by  $i + 1$ .

When the identification of the orbit is complete, the size of the full group, say  $m$ , is known, and the full group is assigned the index of the next group of size  $m$  to be labelled, say  $n$ . Suppose that there are  $k$  positions in the orbit of  $G$ . The current array is then modified as follows: the two arcs of  $G$  are replaced by the label  $m_n^0$ , and

if  $k > 1$ , then for each arc in each pair  $e_i, f_i, i = 1, \dots, k-1$ , the label  $m_n^i$  is inserted between the two group arcs that are joined by the arc in question. This completes the processing of the orbit of  $G$ .

We illustrate this procedure by completing the construction of the master group code for the knot shown in the figure above. We had left off with the control list  $2_3, 2_4, \widehat{-2}_5$ , so the positive group  $2_3$  is the next to be processed. We cycle the current array so as to begin with an arc of group  $2_3$ . The result is:

$$2_3, -2_2^0, 2_4, -2_1^0, \widehat{-2}_5, -2_1^1, -2_2^1, 2_3, -2_2^1, 2_4, -2_1^1, \widehat{-2}_5, -2_1^0, -2_2^0$$

and so  $-2_2^0, 2_4, -2_1^0, \widehat{-2}_5, -2_1^1, -2_2^1$  and  $-2_2^1, 2_4, -2_1^1, \widehat{-2}_5, -2_1^0, -2_2^0$  are the first and second sections, respectively. We start with  $S_0^1$  and  $S_0^2$  empty, and look for the shortest initial subsequences of the first and second sections that contain  $S_0^1$  and  $S_0^2$ , respectively, and which form a closed pair. We find that  $S_1^1$  is the first section and  $S_1^2$  is the second section, so the orbit of  $2_3$  has a single min-tangle. We replace both occurrences of  $2_3$  by  $2_3^0$ , and remove group  $2_3$  from the control list to complete the processing of the orbit of group  $2_3$ . The control list is now  $2_4, \widehat{-2}_5$ , so we cycle the current array so as to begin with an arc of group  $2_4$ , say

$$2_4, -2_1^0, \widehat{-2}_5, -2_1^1, -2_2^1, 2_3^0, -2_2^1, 2_4, -2_1^1, \widehat{-2}_5, -2_1^0, -2_2^0, 2_3^0, -2_2^0,$$

and we find that  $-2_1^0, \widehat{-2}_5, -2_1^1, -2_2^1, 2_3^0, -2_2^1$  and  $-2_1^1, \widehat{-2}_5, -2_1^0, -2_2^0, 2_3^0, -2_2^0$  are the first and second sections. We start with  $S_0^1$  and  $S_0^2$  empty and look for the shortest initial subsequences  $S_1^1$  and  $S_1^2$  of the first and second sections which properly contain  $S_0^1$  and  $S_0^2$ , respectively, and which form a closed pair. We find that  $S_1^1 = -2_1^0, \widehat{-2}_5, -2_1^1$  and  $S_1^2 = -2_1^1, \widehat{-2}_5, -2_1^0$ , so  $e_1$  is the arc that joins  $-2_1^1$  to  $-2_2^1$  in the first section, and  $f_1$  is the arc that joins  $2_1^0$  to  $-2_2^0$  in the second section. At the next step, we find that  $S_2^1$  is the first section, so  $S_2^2$  must be the second section, and the orbit has been completely identified. There are two min-tangles in the orbit of  $2_4$ . We replace both occurrences of  $2_4$  by  $2_4^0$ , and place the label  $2_4^1$  between the first occurrences of  $-2_1^1$  and  $-2_2^1$ , and between the second occurrences of  $-2_1^0$  and  $-2_2^0$ . The result is

$$2_4^0, -2_1^0, \widehat{-2}_5, -2_1^1, 2_4^1, -2_2^1, 2_3^0, -2_2^1, 2_4^0, -2_1^1, \widehat{-2}_5, -2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0.$$

We may now remove group  $2_4$  from the control list to complete the processing of the orbit of group  $2_4$ . The control list is now  $\widehat{-2}_5$ , so we cycle the current array so as to begin with an arc of group  $\widehat{-2}_5$ , say

$$\widehat{-2}_5, -2_1^1, 2_4^1, -2_2^1, 2_3^0, -2_2^1, 2_4^0, -2_1^1, \widehat{-2}_5, -2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0, 2_4^0, -2_1^0$$

with  $-2_1^1, 2_4^1, -2_2^1, 2_3^0, -2_2^1, 2_4^0, -2_1^1$  and  $-2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0, 2_4^0, -2_1^0$  the first and second sections, respectively. We start with  $S_0^1$  and  $S_0^2$  empty and look for the shortest initial subsequences  $S_1^1$  and  $S_1^2$  of the first and second sections which properly contain  $S_0^1$  and  $S_0^2$ , respectively, and which form a closed pair. Since the first arc of the second section is  $-2_1^0$ , and the second occurrence of that group is at the end of the second section, we see that  $S_1^1$  and  $S_1^2$  must be the first and second sections, respectively, and the processing of the orbit of group  $\widehat{-2}_5$  is complete—its orbit has a single min-tangle. We replace both occurrences of  $\widehat{-2}_5$  by  $\widehat{-2}_5^0$ , and remove group  $\widehat{-2}_5$  from the control list to complete the processing of the orbit of group  $\widehat{-2}_5$ . Since the control list is now empty, the construction of the master group code is complete. The master group code that we have constructed from the group code

$$-2_1, -2_2, 2_3, -2_2, 2_4, -2_1, \widehat{-2}_5, 2_3, 2_4, \widehat{-2}_5$$

is

$$\widehat{-2}_5^0, -2_1^1, 2_4^1, -2_2^1, 2_3^0, -2_2^1, 2_4^0, -2_1^1, \widehat{-2}_5^0, -2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0, 2_4^0, -2_1^0.$$

We conclude this section with a description of a method to convert a master group code of the knot that was obtained by the application of  $k - 1$   $\mathcal{D}$  surgeries to a sequence of  $k - 1$  link groups  $G_1, G_2, \dots, G_{k-1}$  in a  $k$ -component link into a master group code for the link.

For each  $i = 1, \dots, k - 1$ , we shall apply  $\mathcal{D}^-$  to  $\widehat{G}_{k-i}$ . The first application of  $\mathcal{D}^-$  will result in a 2-component link, and in general, after the  $i^{\text{th}}$  application, the result is a link of  $(i + 1)$ -components.

Suppose that  $i$ ,  $1 \leq i \leq k - 1$ , is such that we have processed all groups  $\widehat{G}_j$  for  $j < i$ . Thus we have a link of  $i$  components and we are ready to process  $\widehat{G}_i$ , which is a positive component group. We first locate the component that contains  $\widehat{G}_i$  (since  $\widehat{G}_i$  is a positive component group, all of its flype positions lie on this component). If  $\widehat{G}_{k-i}$  has not been flagged (recall that if for some loner  $G$ ,  $\widehat{G}$  was found to be a core tangle in some other negative component group's orbit, then we flagged  $\widehat{G}$  as being a non-flyping group precisely so we would have this information available now), then we cycle the code for this component to bring one 0-position group arc of  $\widehat{G}_{k-i}$  to the front, insert a colon in front of the second 0-position group arc (thereby splitting the component into two components, each containing one group arc of every position pair for  $\widehat{G}_{k-i}$ ), and finally, we replace every occurrence of the label  $\widehat{G}_{k-i}$  by  $|G|_{k-i}$  (two for each position).

Recall that when we were parsing the orbit of a group  $\widehat{G}$ ,  $G$  a loner, if we discovered a trivial flype orbit, we applied  $\mathcal{D}^-$  and then  $\mathcal{D}$  in the orthogonal alignment and parsed for a flype orbit in this direction. We had suggested that it was not actually necessary to perform this pair of surgeries (which involved a reversal of one section of code, an expensive operation), but rather one could parse for the orbit of an imaginary negative 2-group for which the original  $\widehat{G}$  was a core tangle. If a nontrivial orbit is found in this new alignment, then (since in this alternative scheme, we have  $\widehat{G}$  still in its original alignment, a positive 2-group) upon removing the hat we must convert it into a negative (link) loner.

We further remark that when we apply  $\mathcal{D}^-$  to a  $\widehat{G}$ , a positive component group, each of the two components that result from the application of  $\mathcal{D}^-$  will inherit their orientation from that of the component containing  $\widehat{G}$ . It is possible that one of these two components has a different orientation from that of the original code.

We illustrate this algorithm for converting the master group code for the knot into a master group code for the link by completing the calculation for the example we have been developing.

The master group code that we had constructed for the knot of the example was

$$\widehat{-2}_5^0, -2_1^1, 2_4^1, -2_2^1, 2_3^0, -2_2^1, 2_4^0, -2_1^1, \widehat{-2}_5^0, -2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0, 2_4^0, -2_1^0.$$

The sequence of link groups to which  $\mathcal{D}$  was applied has length one, namely  $-2_5$ . The algorithm requires that we cycle the code for the component that contains  $\widehat{-2}_5^0$  to bring one group arc of  $\widehat{-2}_5^0$  to the front. Since there is only one component and its code begins with  $\widehat{-2}_5^0$ , there is nothing to do for this step. We may directly apply  $\mathcal{D}^-$  to  $\widehat{-2}_5^0$  which is achieved by replacing both occurrences of  $\widehat{-2}_5^0$  by  $2_5^0$ . This results in the code

$$2_5^0, -2_1^1, 2_4^0, -2_2^1, 2_3^0, -2_2^1, 2_4^1, -2_1^1 : 2_5^0, -2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0, 2_4^0, -2_1^0$$

which is a master group code for the original 2-component link.

#### 4. The construction of all diagrams for a link

In this section, we describe how every diagram for a link  $L$  can be constructed from a given master group code for  $L$ . The master group code identifies each group and all of its flype positions. From the master group code, extract the subsequence consisting

of all 0-position group labels. This is a full group code for  $L$ , and we call it the *0-position group code* (of the master group code). Let  $D$  be a diagram for the 0-position group code. By Menasco and Thistlethwaite, any diagram  $D_1$  for  $L$  can be obtained from  $D$  from a finite sequence of flypes. As a result of the non-interference of flype orbits, each crossing of  $D$  can be flyped to any of its flype positions, independently of any other flying activity. We construct a set of group codes for  $L$  by the following process. For each full group  $G$  in the diagram  $D$ , we consult the master group code to determine the possible flype positions for the crossings of this group and in all possible ways, we distribute the crossings of this group amongst the various positions. The resulting set of group codes contains at least one code for each diagram for  $L$ .

For example, the 0-position code of the master group code

$$2_5^0, -2_1^1, 2_4^0, -2_2^1, 2_3^0, -2_2^1, 2_4^1, -2_1^1 : 2_5^0, -2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0, 2_4^0, -2_1^0$$

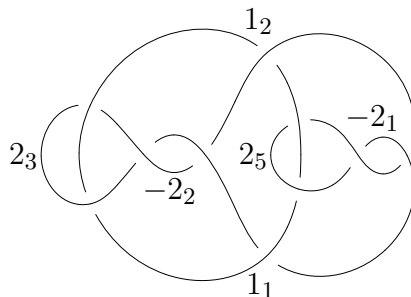
is

$$2_5, 2_4, 2_3 : 2_5, -2_1, -2_2, 2_3, -2_2, 2_4, -2_1.$$

For each of the groups  $-2_1, -2_2, 2_3$  and  $2_5$ , we elect to leave both crossings of the group in 0-position, while for group  $2_4$ , we leave one crossing in the 0-position and flype one crossing to position one of  $2_4$ 's orbit. Since there were no loners initially, we may use labels  $1_1$  and  $1_2$  for these two subgroups of size one. The result is the (split) group code

$$2_5, 1_1, 2_3, 1_2 : 2_5, -2_1, 1_2, -2_2, 2_3, -2_2, 1_1, -2_1$$

whose diagram is shown in Figure 1.



A split-group diagram for the two component link with 10 crossings

Diagram rendered by Knotilus

Figure 1.

## CHAPTER V

### THE MASTER ARRAY

#### 1. The construction of the master array from a master group code

In this section, we introduce the master array for a prime alternating link. The master array is a uniquely determined master group code for the link and it can be constructed from any group code for the link. It is a consequence of this unique determination that the master array is a complete invariant up to mirror image; that is, two link diagrams are flype equivalent, up to mirror image, if and only if the master arrays that are constructed from the two diagrams are identical.

Suppose that we have a master group code for a link of  $n$  crossings and  $k$  components. This is a sequence of  $k$  component group codes, where a component group code is a sequence of labels of the form  $m_j^i$ , where  $m$  is a nonzero signed integer,  $j$  a positive integer and  $i$  is a nonnegative integer. Recall that  $m$  is the size of the group and  $j$  is an index that allows us to distinguish different groups of size  $m$  and  $i$  is the flype position of the group. We shall represent the master group code by a component array  $L[]$  with  $k$  entries. The master group code for each component is represented by a circular linked list (which provides an orientation for the component), and for each  $i := 0$  to  $k - 1$ ,  $L[i]$  is a pointer to the record for the selected starting group arc of the master group code for component  $i$ .

We shall apply a *standardization* procedure to this master group code to obtain another master group code for the link and it is this master group code that is called the master array for the link. It will be seen that when this procedure is applied to any two master group codes for the link, the procedure returns exactly the same master group code.

Effectively, the standardization procedure amounts to ordering the components (that is, we determine which component is listed first, then which is listed second,

and so on), selecting a starting group arc for each component (that is, an entry  $m_j^i$  where  $i$  is not required to be zero), and selecting an orientation for each component. The reader should be alerted to the fact that we will not carry out the steps of the procedure in the order listed above.

The first step in the standardization procedure is to choose the component that will become the first component. To each component, we assign a sort record. We then sort the components according to the sort record with a lexical field-by-field comparison, in each field using either the usual ordering or the reversal of the usual ordering (we indicate the latter situation by writing “reversed”) as described below. There might be more than one component that is lexically least in this comparison, and we shall describe how to deal with this situation shortly. Here is the structure of the sort record.

<code>compendium:</code>	0 if the link has maximum link group size $M > 1$ and this component has a link group of size $M$ , 1 if the link has only loner link groups and, if $N = \max\{ g  \mid g \text{ a negative component group of the link}\}$ (where loners that flype in the negative direction are considered to be groups of size $-1$ ), this component has a negative component group of size $-N$ , $-1$ otherwise.
<code>num_components_met:</code>	the number of components (other than itself) that this component meets
<code>num_loner_link_groups:</code>	the number of loner link groups (counting different positions as different loners) on the component
<code>num_component_groups:</code>	(reversed) the number of component group arcs (all positions) on the component
<code>num_crossings:</code>	(reversed) the number of crossing arcs (all positions) on the component
<code>code_length:</code>	the number of group arcs (all positions) on the component
<code>link_groups_same_size:</code>	a boolean field, true if all link group arcs on the component are the same size, otherwise false
<code>num_component_groups_by_size:</code>	(reversed) an array, initialized to 0, of size $2n + 1$ , where $n$ is the number of crossings of the link. If there are $k$ groups of size $i$ on the component, then if $i < 0$ , we set the entry at index $n -  i $ of this array equal to $k$ , otherwise we set the entry at index $n + i$ equal to $k$ . Recall that loners which flype in the negative direction have size $-1$ , while for this purpose, we include non-flyping loners in the count of groups of size 1. In every case, different positions of the same group are counted as different groups of the same size for this purpose.
<code>num_link_groups_by_size:</code>	(reversed) an array of size $n + 1$ , where $n$ is the number of crossings of the link. The array is initialized to 0. If there are $k$ link groups of unsigned size $i$ , then we set the entry at index $i$ of this array equal to $k$ . Again, we count two different positions of the same link group as different link groups for this purpose. Note also that for this purpose, we do not distinguish between positive and negative link groups.

When the components are sorted according to the above criteria, it is possible that there may be more than one ranked least. In any event, we carry out the following procedure to determine an orientation on each “winner” of the competition; that is, on each component that was ranked least.

The first step in establishing an orientation on a candidate for first component is to identify all longest ladders; that is, subsequences that are without any repetition. A repetition occurs when a group arc is encountered whose matching arc has already appeared in the subsequence under construction (different positions of a group arc are considered to be different groups for this comparison purpose). To find such subsequences in the master group code for the component under consideration, we use two pointers, a tail pointer and a head pointer. Initially, we set the tail pointer in front of the first group arc on the component, and the head pointer is positioned just after the first group arc on the component. As well, we initialize a variable `group_size` to the size of the first group arc. We then advance the head pointer, group arc by group arc, incrementing `group_size` by the unsigned size of each group arc that is passed, until such time as the advance of the head pointer would include a group arc whose matching arc lies between the tail pointer and the head pointer (note that we have not actually advanced the head pointer, but rather we have looked ahead to see what would happen if we did advance it). This ladder’s tail and head pointer positions and the value of `group_size` are entered as the initial entry in the list of ladders of largest size encountered so far. We then position the tail pointer just after the first arc of the group that caused the head pointer to stop its forward advance, and reposition the head pointer to point to the position just after the group arc that follows the tail pointer. Reset the value of `group_size` to the size of the group arc that lies between the tail and head pointer and repeat the above steps, comparing the size of the newest ladder to the previously recorded maximum size. If the new ladder has size larger than those of the previous largest size, discard the list of ladders of the previous largest size and enter our new ladder into the list of ladders of largest size so far. If the size of the newest ladder is the same as the previously recorded greatest ladder size, we add our ladder’s data to the list, move our head and tail pointers to begin the search for the next ladder, which will also reset the `group_size` variable. Finally, if the size of the newest ladder is less than the previously recorded largest ladder size, we discard the ladder, reposition the pointers as above, and continue. When the head pointer reaches the end of the master group code for the component, it simply wraps around and continues from the beginning of the code. The process stops when the tail pointer is advanced past the end of the master group code for the component.

When the list of longest ladders has been compiled for each component that was tied for least, we identify the length of the longest ladder over all of these components, and we discard those components whose longest ladder length was not this maximum value. For each ladder that remains (now working with all longest length ladders from all surviving components), we write the master group code for the component of the ladder from the initial group arc of the ladder through the ladder and on through the entire code until we have reached the arc that immediately precedes the initial ladder arc (remember, the code is cyclic), and we also write it out starting with the last arc of the ladder, through the ladder, wrapping to the end of the code and back until we have reached the arc that immediately follows the last arc of the ladder (or the first arc of the master group code if the last ladder arc was the last arc of the master group code). We then carry out a lexical comparison of these rewritten component master group codes, using early exit to drop any component whose master group code was lexically “larger” than the least at any point in the comparison. For this comparison, the necessary data for each component’s master group code is presented in an array that has one entry per group arc, with each entry being a record with the following structure (in each field, the ordering is either the usual, or else it is the reversal of the usual, and the latter case is indicated by the word “reversed”).

link group or component group:	1 for link group, 0 for component group
group size:	the signed size of the group if a component group (including the sign on a flyping loner), otherwise the unsigned group size.
number of core tangles:	(reversed) 0 is the default value, and this is non-zero only for negative component groups and loners that flype in the negative direction (the number of core tangles is equal to the number of components onto which the group may flype)
orbit size:	(reversed)

The comparison function returns a value of  $-1$ ,  $0$ , or  $1$ , determined as follows:  $-1$  if first record is smaller than the second record,  $0$  if the two are equal, and  $1$  if the first record is larger than the second record. The first field to be examined is the link group or component group field, and we compare these values with the usual ordering. If tied, we are then comparing either two link groups or two component groups.

Case 1: comparing component groups. In this case, we compare their signed sizes again with the usual ordering. If they are equal, then we compare the number of core tangles, with inverted comparison order—that is, the maximum is the winner (if the two groups were positive the core tangle count will be zero for each). If they are still equal, then we compare the flype orbit size, again with inverted comparison order. If they are still tied at this point, we return  $0$ .

Case 2: comparing link groups. In this case, we compare the absolute value of the size field, and if tied, we then compare the flype orbit size with inverted comparison order. If they are still tied at this point, we return 0.

We remark that, unlike link groups, the signs of component groups are not dependent on the orientation of the components. It is for this reason that we do not use the sign of link groups in the comparison function.

The preceding comparison exercise will produce a list of master group codes for the “winning” components; that is, those that are tied for lexically least. For each of these winning component master group codes, we construct a data structure which is capable of holding a complete master group code for the link, but which has at this stage its first component master group code set to be the master group code for the winning component under consideration and the remaining entries in the data structure are set to be null. Each of these data structures shall be referred to as a branch of the standardization process. It will be necessary to store additional information for each branch of the standardization process and so we augment the data structure for the storage of the various component master group codes with additional data fields.

Here is the data structure for the complete branch record.

```

components: an array of size number_of_components. For each  $i$ , position  $i$  of
this array contains a record of data for component  $i$ , where the
record structure is:
    met: default is 0, set to a nonzero value of  $k$  when the
component has been first encountered, where
 $k - 1$  is the number of components that have
already been met at this point.
    direction: default is 0, set to a nonzero value at the time
this component is oriented by this branch of the
standardization scheme. When oriented, the
direction entry is set to 1 if oriented in the
same direction as in the original master group
code,  $-1$  if oriented in the opposite direction.
    head: a pointer to the record for the selected start-
ing group arc of the master group code for the
component.
order_oriented: this is a list of component indices (0 to  $k - 1$ ). As the branch
of the standardization process orients a component, the index
of that component in the array of components for the original
master group code is added to the end of this list.
met_but_not_oriented: this is a list of components. Each time this branch of the stan-
dardization process identifies a component but is unable to orient
it, the index of that component in the array of components for
the original master group code is added to the end of this list.
num_met_but_not_oriented: the number of entries in the list met_but_not_oriented.

```

Each branch record is initialized as follows. For each component record in the `component` array for the branch, we set the `met` and the `direction` field to 0 and the `head` field is set to null. Then, if the master group code of the component that has been selected to be the first component of this branch has the same orientation as it had in the original master group code, we set the `met` field to be 1, set the `direction` field for this component to 1 (specifically, if this component has index  $i$  in the original master group code, then we set `components[i].direction = 1`), otherwise set it to  $-1$ , and set the `head` field to point to the group arc record for the component master group code.

Initialize `order_oriented` to a list with one entry; namely  $i$ , the index of the first component of this branch as it appeared in the original master group code. As well, set `num_of_o_components := 1`. Initialize `met_but_not_oriented` to an empty list, and set the value of `num_met_but_not_oriented := 0`. Set `o_oriented` equal to 0 (`o_oriented` points to the current entry in the list `order_oriented` that is driving the standardization process, and initially it points to the one and only entry in the list for the branch).

The branches are organized as a linked list, and `standardization_head` points to the first branch in the list, `standardization_current` points to the current branch being processed, initially the first branch in the list, `standardization_end` points to the last entry in the list. `standardization_component_position` is the index of the link group arc that is currently being processed (the same across all branches) on the component selected from the branch's list of oriented components by the value of `o_oriented`, and it is initially set to the value of `head` for the selected component.

For this value of `standardization_component_position` we iterate through all branches, performing a comparison based on a function called `attempt_to_orient`. The comparison and the `attempt_to_orient` function will be described shortly. Once a full pass of all branches has been made, `standardization_component_position` is advanced to point to the next link group on the component pointed at by `o_oriented` and repeat the iteration across all branches. Once all link groups on the component `o_oriented` have been processed, we advance `o_oriented` to point to the next component in the list of oriented components, or, if that list is empty, we use the first component in the `met_but_not_oriented` list (for each branch, since the way the procedure is carried out will guarantee that all branches will have emptied the `order_oriented` list at the same time) to set up two new branches with this newest component oriented each of the two possible ways (so this causes a doubling of the number of branches). This will require that we remove the component

from the list of `met_but_not_oriented` components and move it to the list `oriented_components`, advancing `o_oriented` to point to it. Set the value of `standardization_component_position` is set to point to the link group arc which was the link group that resulted in the component first being met, and continue as above. Note that if the list `order_oriented` is exhausted, there will be components in the list `met_but_not_oriented`. The process stops when every component has been oriented.

We now describe the comparison process and the function `attempt_to_orient` that is used by the comparison process. For the current value of `standardization_component_position`, we carry out the following steps for the first branch; that is, the branch pointed to by `standardization_head`. Let `head_arc` point to the other group arc of the link group at position `standardization_component_position` on the component selected from the branch's list of oriented components by the value of `o_oriented` and let `head_component` denote the index (in the original master group code) of the component that contains `head_arc`.

We first check to see if the component at index `head_component` in the list of components for the branch has been met. If not, then we set the value of its `met` field to  $k + 1$ , where  $k$  is the number of components met so far, and we set the value of its `head` field to point to the link group arc at position `standardization_component_position`; that is, set

```
standardization_head.components[head_component].met := k + 1 and
standardization_head.components[head_component].head := head_arc.
```

Next, we check to see if `head_component` has been oriented. If not, then we apply the `attempt_to_orient` function to `head_component`.

The `attempt_to_orient` process takes two arguments: the (index of the) component  $X$  it is to attempt to orient, and `head_arc`, and it returns  $+1$ ,  $-1$  or  $0$  if  $X$  was, respectively, able to be oriented and the orientation is the same as that of  $X$  in the original master group code, able to be oriented and the orientation is not the same as that of  $X$  in the original master group code, or not able to be oriented.

Case 1:  $X$  has just been met for the first time and either  $|G| > 1$  or  $G$  is a flying loner, or  $X$  has been met for the second or third time and  $|G| > 1$ . Then `attempt_to_orient` returns the product of the sign of  $G$  in the original master group code and the value of the `direction` field for the component `standardization_head.order_oriented[o_oriented]`. Note that this will cause  $G$  to be positive group in the resulting master group code.

Case 2:  $G$  is a loner and either  $G$  is non-flying or  $X$  has been met before. In a temporary work area, we write the master group code for  $X$  in both +1 and -1 directions, in each case with starting point  $G$  if  $X$  has not been met before, otherwise ( $X$  has been met before) with starting point the group arc pointed to by the value of the `head` field in  $X$ 's record, and, using the rules as presented in Lexical Comparison 1, we perform a lexical comparison of the two component master group codes, with the first and second arguments being the +1 direction and the -1 direction, respectively.

**V.1.1 LEXICAL COMPARISON.** The rules, listed in order of application, for determining that  $m_i^j < n_r^s$  are:

- (i)  $m_i^j$  is a component group and  $n_r^s$  is a link group.
- (ii) both are link groups and either  $|m| < |n|$ .
- (iii) both are component groups and  $m < n$ , or  $m = n$  and the number of core tangles of group  $m_i$  is greater than the number of core tangles of group  $n_r$  (the number of core tangles of a positive group is equal to zero).
- (iv) the number of flype positions for  $m_i$  is greater than that for  $n_r$
- (v) the value of `met` for the component containing the other arc of  $m_i^j$  is less than that of  $n_r^s$  if they are both non zero, or if the value for  $m_i^j$  is non zero and that of  $n_r^s$  is 0.
- (vi)  $X$  has been met before and  $m_i^j$  is the group arc pointed to by  $X$ .`head`, and  $n_r^s \neq m_i^j$ .

If not tied, then we return the orientation, +1 or -1 of the winning component master group code, otherwise we return 0

We remark that the `attempt_to_orient` process guarantees that by the third time a component has been encountered, it will be oriented.

We now assign, a value to a variable `head_compare` as follows. If  $X$  was already oriented, then we set `head_compare` equal to 3. On the other hand suppose that  $X$  was not already oriented, so that we will have run the `attempt_to_orient` process. There are four possible outcomes, and we assign the value to `head_compare` as described below.

$$\text{head\_compare} = \begin{cases} 3 & X \text{ was not able to be oriented.} \\ 2 & X \text{ has been met for the first time and the } \text{attempt\_to\_orient} \\ & \text{process was not able to orient } X. \\ 1 & X \text{ has been met for the second time and the } \text{attempt\_to\_orient} \\ & \text{process was able to orient } X. \\ 0 & X \text{ has been met for the first time and the } \text{attempt\_to\_orient} \\ & \text{process was able to orient } X. \end{cases}$$

If `head_component` has been met before and oriented now, it must be removed from the list of components met but not oriented for the first branch and added to the end of the list of oriented components for the first branch, while if the component has just now been met and was oriented, it is simply added to the end of the list of oriented components. If the component has been met before, but was unable to be oriented, then no action is taken, while if it has just now been met but was unable to be oriented, it is added to the end of the list `met_but_not_oriented`.

Now, (still with the current value of `standardization_component_position`, we set `standardization_current = standardization_head` and we initialize a boolean variable `more_branches` to false if there is only one branch, true otherwise. Now we perform the following while loop.

While (`more_branches`), advance `standardization_current`.

If the branch now pointed to by `standardization_current` is the last branch, set the value of `more_branches` to false. Let `current_component` denote the index (in the original master group code) of the component that contains the other group arc of the link group arc at position `standardization_component_position` of the component

`standardization_current.order_oriented`.

We assign a value to a variable `current_compare` exactly as was done above for `head_component` (the only change required in the description of the calculation is to replace `standardization_head` by `standardization_current` and `head_compare` by `current_compare`).

If the `current_component` has been met before and oriented now, it must be removed from the list of components met but not oriented for the current branch and added to the end of the list of oriented components for the current branch, while if the component has just now been met and was oriented, it is simply added to the end of the list of oriented components. If the component has been met before, but was unable to be oriented, then no action is taken, while if it has just now been met but was unable to be oriented, it is added to the end of the list `met_but_not_oriented`.

If `current_compare` is greater than `head_compare`, then the current branch is removed from the list of branches, while if `current_compare` is less than `head_compare`, we set `standardization_head` set equal to `standardization_current`, which causes all branches that precede the current branch to be pruned. Otherwise, it must be that `head_compare=current_compare` and we fall into exactly one of the following four cases, based on the value of `head_compare`:

Case 0: To compare the components at position `o_oriented` in the `order_oriented` list of the branches `standardization_head` and `standardization_current`, each component with their newly assigned directions and starting positions as recorded in their respective `head` fields, we use the lexical comparison criteria as given in Lexical Comparison 1, with the exception of condition (vi). If `standardization_head` is the winning branch, then we remove `standardization_current` and continue, while if `standardization_head` is the losing branch, then we set `standardization_current` to be `standardization_head`. If they tie, continue.

Case 1: We use the `met` value for the current component in `standardization_head` and that for `standardization_current`, choosing the branch with the least as in case 0. This ensures that these component were met at the same time and at that time were lexically equal (it is true that due to updated `met` values, a lexical distinction might now be possible but we don't check for it).

Case 2: To compare the components at position `o_oriented` in the `order_oriented` list of the branches `standardization_head` and `standardization_current`, each component with the direction that of the original master group code and starting positions as recorded in their respective `head` fields, we use the lexical comparison criteria as given in Lexical Comparison 1, with the exception of condition (vi). Since we have just now compared each component lexically to itself in both directions from the group arc by which the component has just now been met, using all six rules as listed in Lexical Comparison 1 and a tie was declared, this means that if we were to lexically compare each component to itself in both directions from the head group arc, using all six rules as listed in Lexical Comparison 1, the result would be a tie. Thus for either component, it does not matter which direction we choose to use in the comparison. If `standardization_head` is the winning branch, then we remove `standardization_current` and continue, while if `standardization_head` is the losing branch, then we set `standardization_current` to be `standardization_head`. If they tie, continue.

Case 3: Continue.

End of while loop.

When all components have been oriented, we create a copy of the original master group code for each branch. Then we work through each branch, processing the components in the order that they appear in the `order_oriented` list for the branch. We traverse each component from its `head` group arc in the direction determined by the value of the `direction` field (+1 means in the original orientation of the

component while  $-1$  means in the opposite orientation). For each positive integer  $m$  for which the master group code contains a group whose size in magnitude is equal to  $m$ , we create a list, initially empty, in which we will record information about each group, as it is encountered, whose size, in magnitude, is equal to  $m$ . Each record in the list for an integer  $m$  will have the form

**index:** equal to  $k$  if this is the entry for group  $\pm m_k$ .  
**flype\_position\_list:** this points to the first entry in the list of flype positions for this group.

Each record in the list of flype positions for a group has the following form.

**arc\_one:** points to the first group arc of this flype position, as encountered during the traversal of the branch.  
**arc\_two:** points to the second group arc of this flype position, as encountered during the traversal of the branch.  
**next:** points to the next flype position record, set to null if this is the last flype position record.

As the branch is traversed, for each group arc label  $\pm m_i^j$  that is encountered, we consult the list for integer  $m$  to see if index  $i$  has yet been recorded. If not, then we create a new flype position list entry with the **arc\_one** field to point to the position of the group arc we have just encountered, and we append to the end of the list a record for index  $i$ , setting its **flype\_position\_list** field to point to the newly created flype position record. On the other hand, if index  $i$  does appear, then we read through the entries in the flype position list, examining the flype index of the group arc pointed to by **arc\_one** for each entry, searching to see if flype index  $j$  has already appeared. If not, then we create a new flype position list entry with the **arc\_one** field to point to the position of the group arc we have just encountered, and we set the value of **next** field in the last record of the flype position list to point to the newly created flype position record. Finally, if we find an entry in the flype position list for which **arc\_one** is equal to  $j$ , then we set the value of **arc\_two** for that entry to point to the current group arc label.

Once the entire branch has been traversed, for each positive integer  $m$  for which a list has been created, we relabel, in the branch's copy of the original master group code, all groups that appear in this list (and their flype positions) as follows. We work our way through the list from beginning to end, and if we are processing the  $i^{\text{th}}$  entry in the list (the first entry corresponds to  $i = 1$ ), then we work through the flype position list for this entry. If we are processing entry  $j$  in the flype position list ( $j = 0$  corresponds to the first entry in the list), then in the group arc record pointed to by both **arc\_one** and **arc\_two**, change the group index label  $i$  and the flype position index to  $j$ .

The final step in the standardization process is to perform a lexical comparison between the master group codes that have been constructed from the branches. For the purpose of this comparison, entry  $m_i^j < n_r^s$  if

- (i)  $m_i^j$  is a component group and  $n_r^s$  is a link group.
- (ii) both are link groups or both are component groups, and either  $m < n$  or  $(m = n$  and  $i < r)$  or  $(m = n, i = r,$  and  $j < s)$ .

All lexically least master group codes from this comparison will be identical, and each is the master group code that we call the master array for the link (and its mirror).

When this standardization algorithm is applied to the master group code that we constructed earlier for a 10 crossing, 2-component link,

$$2_5^0, -2_1^1, 2_4^0, -2_2^1, 2_3^0, -2_2^1, 2_4^1, -2_1^1 : 2_5^0, -2_1^0, 2_4^1, -2_2^0, 2_3^0, -2_2^0, 2_4^0, -2_1^0,$$

the result is

$$2_1^0, -2_2^0, 2_3^0, -2_4^0, 2_5^0, -2_4^0, 2_3^1, -2_2^0 : 2_1^0, -2_2^1, 2_3^0, -2_4^1, 2_5^0, -2_4^1, 2_3^1, -2_2^1,$$

which is therefore the master array for the link.

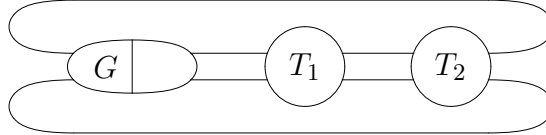
## 2. Bounds on the size of master group codes and the number of diagrams

In this the final section we show that the size of the master array for a prime alternating link  $L$  of minimal crossing number  $n$  is at most  $4(n - 3)$ , and that the maximum number of minimal diagrams (not necessarily unique) that could be constructed from the master array is  $2^{n-4}$ . Furthermore, we give examples to show that both of these bounds can be achieved.

**V.2.1 PROPOSITION.** *The master array of a prime alternating link of minimal crossing size  $n \geq 4$  has at most  $4(n - 3)$  group arcs.*

*Proof.* The only prime alternating links of 4 crossings are the figure-eight knot and the four crossing torus link. The master array of the figure-eight knot has two group arcs for each of its negative component groups of size two so a total of four, while the master array of the four crossing torus link has only two group arcs for its link group of size four. Since  $2 < 4 \leq 4(4 - 3)$ , the claim holds for  $n = 4$ . We remark that the only prime alternating links with less than four crossings is the Hopf link and the trefoil both of which have a master array of size two. Suppose now that  $n > 4$

is an integer such that for all prime alternating links of at least 4 and at most  $n - 1$  crossings, the claim holds, and let  $L$  be a prime alternating link of  $n$  crossings. If there are no flying groups in  $L$ , then the master array for  $L$  has at most  $2n$  group arcs achieved if every full-group of  $L$  is a loner group, where  $4(n - 3) = 2n + 2n - 3 \geq 2n$  since  $n \geq 5$ . Suppose then that  $L$  has a flying group  $G$  of size  $c$ . Let  $D$  be a minimal diagram of  $L$ , and let  $T_1$  be a min-tangle in the orbit of  $G$ , and let  $T_2$  denote the 4-tangle that comprises the rest of the orbit of  $G$ , so that  $D$  can be represented as shown below.



We need to determine the number of group arcs that lie on the two strands of the 4-tangle  $T_1$ , and the number of group arcs that lie on the two strands of the 4-tangle  $T_2$ , since the sum of these two numbers plus 2 for each of the two positions of  $G$  that lie outside of both  $T_1$  and  $T_2$  is the number of arcs in the master array of  $L$ . Since no group inside the min-tangles  $T_i$ ,  $i = 1, 2$ , can flype onto a pair of arcs outside of  $T_i$  by Theorem III.1.10, it suffices to determine the number of arcs in the master array of



where it should be noted that we have replaced the group  $G$  by a group of size 2 so that the 2-group is not a min-tangle in the flype orbit of  $G$ . This will preserve the orbit structure of those groups in either  $T_1$  or  $T_2$  that flypes over a min-tangle that includes  $G$ . The two positions of  $G$  that lie outside of  $T_1$  and  $T_2$  are now accounted for by the two copies of the group of size two, so the number of arcs in the  $MA$  of  $L$  is the sum of the number of arcs in the master array of the two prime alternating links whose diagrams are as shown above. Note that each of these two diagrams is still a reduced alternating diagram, and that each of  $T_1$  and  $T_2$  must have at least two crossings by definition of a min-tangle, so each of the two links has at least 4 crossings and no more than  $n - 1$  crossings even if  $|G| = 1$  since there is at least two min-tangles. Note that we chose a group of size two to replace  $G$  (even if  $G$  was a loner) as opposed to a single crossing to ensure we don't drop below our base case size. Two other important reasons for this choice is that a loner has the potential to flype along either alignment which means we may not preserve the orbit structure of  $G$  or a flying group inside  $T_1$  that used the position arcs of  $G$  on each side of the

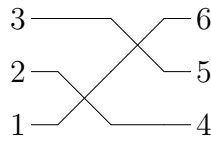
min-tangle which means the replacement loner would amalgamate with the flying group reducing the size of the orbit of this group by one. Secondly we could possibly create more flype positions inside the min-tangle  $T_1$  or inside  $T_2$  if it was a min-tangle as well. In other words the min-tangle  $T_1$  (and if  $T_2$  is a min-tangle) may be a nontrivial flype orbit for the replacement crossing in the direction orthogonal to the flype orbit of  $G$  in either of the two links constructed. Since we prevent all these scenarios from happening while preserving the flype structure of  $G$  we can apply the inductive hypothesis to each. Let  $n_1$  denote the number of crossings in  $T_1$ . Then the number of group arcs in the master array of  $L$  is at most

$$4(n_1 + 2 - 3) + 4(n - n_1 - c + 2 - 3) = 4(n - c - 2).$$

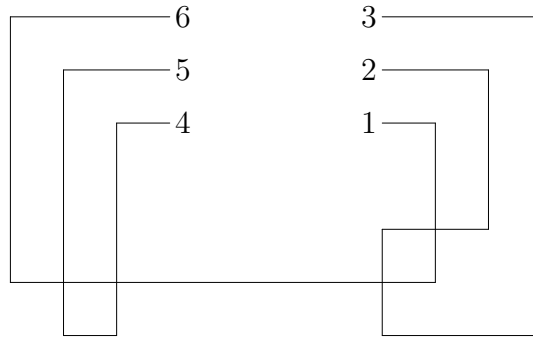
Since  $c \geq 1$ , it follows that  $L$  has at most  $4(n - 3)$  arcs in its master array. The result follows now by induction.  $\blacksquare$

In fact, this bound is best possible. It is achieved by every member of the extreme flyper family (of which the smallest is the figure-eight knot).

**V.2.2 EXAMPLE.** The extreme flyper family is parameterized by a non-negative integer  $k$ , where the  $k^{\text{th}}$  member is constructed by inserting  $k$  copies of the 3-braid

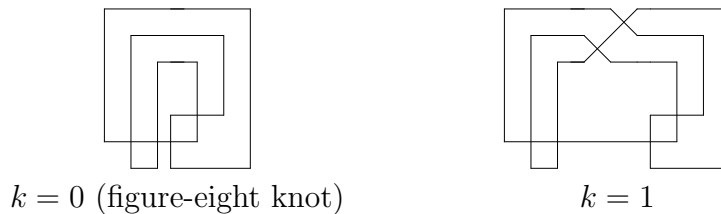


into the base structure

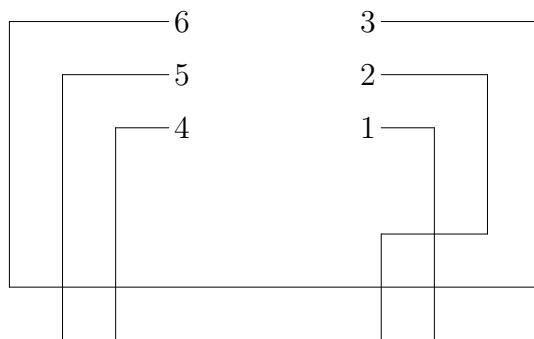


where we connect 1 and 4, 2 and 5, and 3 and 6 on consecutive 3-braids, as well as to connect a 3-braid to the structure.

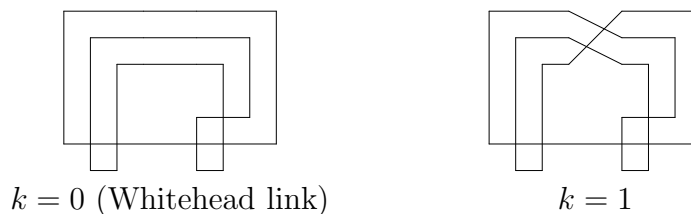
For example,



For the family of extreme flypers with odd size  $n$  we have the base structure



For example,



For  $k \geq 0$ , the  $k^{\text{th}}$  entry in the extreme flyper family of even  $n$  has  $2k$  flying loners and 2 non-flying groups of size two. The orbit of each loner has two positions, so each loner contributes 4 group arcs to the master array. The two 2-groups together contribute 4 group arcs to the master array, so the master array has  $4(2k) + 4$  group arcs. Since the number of crossings  $n$  is given by  $n = 2k + 4$ , we see that the master array has  $4(n - 4) + 4 = 4(n - 3)$  group arcs.

While for  $k \geq 0$ , the  $k^{\text{th}}$  entry in the extreme flyper family of odd  $n$  has  $2k + 1$  flying loners and 2 non-flying groups of size two. The orbit of each loner has two positions, so each loner contributes 4 group arcs to the master array. The two 2-groups together contribute 4 group arcs to the master array, so the master array has  $4(2k + 1) + 4$  group arcs. Since the number of crossings  $n$  is given by  $n = 2k + 5$ , we see that the master array has  $4(n - 5 + 1) + 4 = 4(n - 3)$  group arcs.

Let  $L$  be a prime alternating link and let  $D$  be a reduced alternating full-group diagram for  $L$ . Let  $r$  denote the number of groups in  $D$  and label the groups of  $D$  in an arbitrary fashion as  $g_i$ ,  $1 \leq i \leq r$  and use this labelling to write out a group

code  $C$  for the diagram and from that, construct a master group code for  $L$ . For each group  $g_i$ , let  $t_i$  denote the number of crossings in the group and let  $c_i$  denote the number of flype positions for the group in  $D$ . Now, the number of ways to split group  $g_i$  into subgroups and distribute the subgroups around flype orbit of  $g_i$  (where we do not distinguish between two subgroup of  $g_i$  of equal size) is  $\binom{c_i+t_i-1}{t_i-1}$ . Thus the number of group codes for  $L$  that can be formed from  $C$  in this manner is given by  $\prod_{i=1}^k \binom{c_i+t_i-1}{t_i-1}$ . Since this depends only on  $L$  and not on any particular minimal diagram of  $L$ , we shall denote this number by  $N(L)$ . Thus  $N(L)$  can be computed from any master group code for  $L$ . Specifically, if the full-group had  $r$  groups  $g_i$ ,  $i = 1, \dots, r$ , with  $g_i$  of size  $c_i$  and having orbit size  $t_i$ , then

$$N(L) = \prod_{i=1}^r \binom{c_i+t_i-1}{t_i-1}.$$

Since the number of different diagrams of a prime alternating link  $L$  is at most  $N(L)$ , our initial objective will be met if we establish that  $N(L)$  is at most  $2^{n-4}$ , where  $n$  is the number of crossings in a minimal diagram for  $L$ .

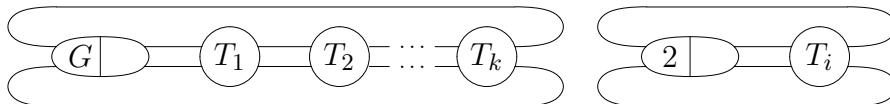
**V.2.3 PROPOSITION.** *Let  $L$  be a prime alternating link with an  $n$  crossing diagram,  $n \geq 4$ . Then  $N(L) \leq 2^{n-4}$ , and so  $L$  has at most  $2^{n-4}$  minimal diagrams.*

*Proof.* The proof is by induction on  $n \geq 4$ . If  $L$  is a prime alternating link with a diagram  $D$  with four crossings, then either  $D$  is not minimal, in which case a minimal diagram for  $L$  has at most 3 crossings which is only possible for the unknot, the Hopf link, or the trefoil, or  $D$  is minimal, in which case  $L$  is either the figure-eight knot or the four crossing torus link. In all cases above,  $N(L) = 1$ .

Suppose now that  $n > 4$  is such that for every alternating link of minimal crossing size  $m$  with  $4 \leq m < n$ ,  $N(L) \leq 2^{m-4}$ , and consider a prime alternating link  $L$  of minimal crossing size  $n$ .

Let  $D$  be a minimal diagram for  $L$  in full-group. If every group of  $D$  has a trivial flype orbit, then  $N(L) = 1$ . Suppose then that  $L$  has at least one flying group  $G$ . Let  $c$  denote the number of crossings in  $G$  and let  $k \geq 2$  denote the size of the flype orbit of  $G$ .

We may represent  $D$  in the form



where the tangles  $T_1, \dots, T_k$  are the min-tangles of the orbit of  $G$ . For each  $T_i$ ,  $i = 1, 2, \dots, k$ , form the prime alternating link  $L_i$  with minimal diagram  $D_i$

Denote the number of crossings in the min-tangle  $T_i$  by  $n_i$ . By the definition of min-tangle,  $n_i \geq 4$  and so each link  $L_i$  has a minimal diagram with  $m_i = n_i + 2 \geq 4$  crossings. Since there are at least two min-tangles in the orbit of  $G$ , it follows also that  $m_i < n$ , and so we may apply the induction hypothesis to each  $L_i$  to conclude that  $N(L_i) \leq 2^{m_i-4}$ . Furthermore, since no group inside a min-tangle  $T_i$ ,  $i = 1, 2, \dots, k$ , can flype onto a pair of arcs outside of  $T_i$  by Theorem III.1.10, we see that all groups of  $T_i$  and their orbit sizes are the same in  $D_i$  as they were in  $D$  (it is important to note that replacing  $G$  by a crossing might create a scenario where the crossing could now lie in the orbit of some group of  $T_i$ ). Thus

$$\begin{aligned} N(L) &= \binom{c+k-1}{k-1} \prod_{i=1}^r N(L_i) \leq \binom{c+k-1}{k-1} \prod 2^{m_i-4} \\ &= \binom{c+k-1}{k-1} 2^{\sum_{i=1}^k (n_i+2-4)} = \binom{c+k-1}{k-1} 2^{n-c-2k}. \end{aligned}$$

It will suffice to prove that  $\binom{c+k-1}{k-1} 2^{n-c-2k} \leq 2^{n-4}$ ; equivalently, that

$$\binom{c+k-1}{k-1} \leq 2^{2k+c-4} = 2^{c+k-1+(k-3)}.$$

This is immediate for  $k \geq 3$ , and since  $k \geq 2$ , it remains only to verify the case when  $k = 2$ ; that is, that  $c \leq 2^c$ , which is true. The result follows now by induction. ■

This is the best possible bound as the family of extreme flypers of both even and odd crossing size  $n$  achieve this bound.

## CHAPTER VI

### CONCLUSIONS

We have presented an effective procedure for the calculation of a complete invariant for prime alternating links. This invariant, called the master array, can be calculated from any alternating diagram for the link.

#### 1. Complexity

While our procedure is effective, we have not yet determined its complexity. The first stage in the construction of the master array is to build a master group code from a gauss or group code for the link, and this computation takes time linear in the number of crossings. However, the standardization algorithm that converts a master group code into a master array may very well take time exponential in the number of crossings. To be more precise, if the list of oriented components is exhausted before every component for the link is oriented, then we take the first component in the list of met but not oriented components and write the component master group code for this component in both directions from the group arc that was first encountered. This causes a doubling in the number of branches being processed, and hence the possibility for exponential blow-up. However, we have implemented these algorithms and we have generated the master arrays for all prime alternating links up to 23 crossings inclusive. During this process, we watched for this situation to occur and we found that only a handful of links exhibited this behaviour. Moreover, in every case (beginning with the Borromean link), it was only when processing the very first oriented component that the list was emptied. Based on our examination of the problematic links, we feel that it will always be the case that after the second component of the link has been oriented, the list of oriented components will never again be exhausted. However, this remains to be proven.

We remark that one occasionally finds remarks in the literature to the effect that flype equivalency between two alternating diagrams for an alternating link is evident

from their alternating diagrams. However, it should be appreciated that if we were given two different group codes for one of the links from the extreme flyper family (see Example V.2.2), then determining equivalency by flyping one group at a time will be infeasible due to the fact that the number of flype choices is exponential in the number of crossings.

## 2. Enumeration

As we have suggested, the master array is of primary importance for enumeration. In fact, we have used the master array with great success to enumerate all prime alternating links of at most 23 crossings, taking 19 days on a desktop computer for the task. Not only that, but less than 50kB of main memory and three terabytes of disk storage were required. The entire database (98,517,495,461 links in all) can be accessed online at Knotilus [11].

## 3. Composite links

It is natural to ask whether the master array can be adapted to give a complete invariant for all alternating links. While one would never enumerate composite links, it is nevertheless an interesting question, and one that can be readily answered as follows. It is well known that every alternating link can be written as a sum of prime alternating links, with the sum unique up to the order of the summands. Given a group code for an alternating link, the group code for each summand can be extracted, and then from each summand's group code, the master array for the summand can be constructed. It should be observed that in forming the summation, writhe must be taken into consideration. For example, the granny knot of six crossings is a composite of one left hand trefoil and one right hand trefoil, whereas the square knot is a composite of two right handed trefoils or two left handed trefoils. Thus we cannot simply represent this composite knot by listing two copies of the trefoil's master array. It will therefore be necessary to use some form of extended master array, by which we mean the master array together with the writhe information for each group. Since we are only dealing with alternating links (oriented by the master array), specifying the writhe of one group determines the writhe of every group and we will choose to assign positive writhe to the first group in the master array. Since we are working up to mirror image, we will have free choice for the specification of writhe of the first group for the master array for one of the summands. In order to uniquely determine a "master array" for the link, we require a convention to uniquely

order the list of master arrays, one for each summand. Once the list of master arrays has been ordered, our convention will be to specify positive writhe for the first group of the first master array. For example, the master array for the granny knot would be  $3_1^0, 3_1^0 \# 3_1^0, 3_1^0(-)$ , while the master array for the square knot would be  $3_1^0, 3_1^0 \# 3_1^0, 3_1^0$ .

We offer one possible convention for the ordering of the list of master arrays for a composite link. The first criteria would be to have prime links of smaller crossing size precede those of larger size. Then among the prime summands of the same crossing size, order their master arrays, lexicographically.

## Bibliography

- [1] F. Bonahon, L. Siebenmann, The classification of algebraic links, unpublished manuscript.
- [2] J. A. Calvo, Knot enumeration through flypes and twisted splices. *J. Knot and its Ram.*, 6(1997), no. 6, 785–798.
- [3] J. H. Conway, An enumeration of knots and links, and some of their algebraic properties, 1970 Computational Problems in Abstract Algebra (Proc. Conf., Oxford, 1967) pp. 329–358 Pergamon, Oxford
- [4] Helmut Doll and Jim Hoste, A tabulation of oriented links, *Math. Comp.* 57 (1991), no. 196, 747–761.
- [5] C. H. Dowker and Morwen B. Thistlethwaite, Classification of knot projections, *Topology Appl.* 16 (1983), no. 1, 19–31.
- [6] K. F. Gauss, Zur mathematischen Theorie der electrodynamischen Wirkungen, *Werke Konigl. Gesell. Wiss. Gottingen* 5:605, 1883.
- [7] J. Hoste, B. Arnold, M. Au, C. Candy, K. Erdener, J. Fan, R. Flynn, R. Muir, and D. Wu, Tabulating alternating knots through 14 crossings, *J. Knot Theory Ram.* 3 no. 4 (1994), 433-437.
- [8] J. Hoste, M. Thistlethwaite, and J. Weeks, The First 1,701,936 Knots, *Math. Intelligencer* 20(1998), no. 4, 33–48.
- [9] Louis H. Kauffman, State models and the Jones polynomial, *Topology* 26 (1987), no. 3, 395–407.
- [10] T.P. Kirkman, The enumeration, description, and construction of knots of fewer than 10 crossings, *Trans. Roy. Soc. Edinburgh* 32 (1883-4), 281-309.
- [11] Knotilus, <http://knotilus.math.uwo.ca>.
- [12] J. B. Listing, *Vorstudien zur Topologie*, Goettinger Studien (Abtheilung 1), 1 (1847), 811–875.
- [13] C. N. Little, On knots, with a census for order 10, *Trans. Connecticut Academy Sci.* 18, (1885), 374–378.
- [14] C. N. Little, Alternate +/- knots of order eleven, *Trans. Royal. Soc. Edinburgh* 36 (1890-1), 253-255.

- [15] William Menasco and Morwen Thistlethwaite, The classification of alternating links, *Ann. of Math.*, (2) 138 (1993), no. 1, 113–171.
- [16] Kunio Murasugi, Jones polynomials and classical conjectures in knot theory, *Topology* 26 (1987), no. 2, 187–194.
- [17] K. A. Perko, On the classification of knots, *Proc. Amer. Math. Soc.*, 45 (1974), 262–266.
- [18] K. A. Perko, Primality of certain knots, *Topology Proceedings* 7, (1982), 109–118.
- [19] Stuart Rankin, Ortho Flint, and John Schermann, Enumerating the prime alternating knots I, *J. Knot Theory Ramifications* 13 (2004), no. 1, 57–100.
- [20] Stuart Rankin, Ortho Flint, and John Schermann, Enumerating the prime alternating knots II, *J. Knot Theory Ramifications* 13 (2004), no. 1, 101–149.
- [21] Stuart Rankin, Ortho Flint, Enumerating the prime alternating links, *J. Knot Theory Ramifications* 13 (2004), no. 1, 151–173.
- [22] N. J. A. Sloane, The On-Line Encyclopedia of Integer Sequences, [www.research.att.com/~njas/sequences/](http://www.research.att.com/~njas/sequences/).
- [23] P. G. Tait, On knots, *Trans. Roy. Soc. Edinburgh* 28 (1877), 145–190.
- [24] P. G. Tait, On knots II, *Trans. Roy. Soc. Edinburgh* 32 (1883-4), 327-342.
- [25] P. G. Tait, On knots III, *Trans. Roy. Soc. Edinburgh* 32 (1884-5), 493-506.
- [26] Morwen B. Thistlethwaite, A spanning tree expansion of the Jones polynomial, *Topology* 26 (1987), no. 3, 297–309.
- [27] L. B. Treybig, A Characterization of the Double Point Structure of the Projection of a Polygonal Knot in Regular Position, *Trans. Amer. Math. Soc.*, Vol. 130, no. 2. (1968), 223-247.

## Vita

Name: Ortho Flint, aka Dwayne K. Smith

Post-secondary Education and Degrees: The University of Western Ontario  
London, Ontario, Canada  
2002-2003 M. A.

The University of Western Ontario  
London, Ontario, Canada  
2003-2007 Ph. D.

Honours and Awards: National Science and Engineering Research Council  
Post Graduate Scholarship (Masters)  
2002-2003

National Science and Engineering Research Council  
Post Graduate Scholarship (Ph. D.)  
2003-2004

National Science and Engineering Research Council  
Canada Graduate Scholarship (Ph. D.)  
2005-2007

Faculty of Science Graduate Student Teaching Award  
The University of Western Ontario, 2004

Faculty of Science Graduate Student Teaching Award  
The University of Western Ontario, 2005

Related Work Experience: Teaching Assistant  
The University of Western Ontario  
2002-2006

Lecturer  
The University of Western Ontario  
2007

### Publications:

Stuart Rankin, Ortho Flint, and John Schermann, Enumerating the prime alternating knots I, *J. Knot Theory Ramifications* 13 (2004), no. 1, 57–100.

Stuart Rankin, Ortho Flint, and John Schermann, Enumerating the prime alternating knots II, *J. Knot Theory Ramifications* 13 (2004), no. 1, 101–149.

Stuart Rankin, Ortho Flint, Enumerating the prime alternating links, *J. Knot Theory Ramifications* 13 (2004), no. 1, 151–173.