Patrick David Bangert

Algorithmic Problems in the Braid Groups

Submitted for the degree of Ph.D.
Supervisor: Prof Mitchell A. Berger

Department of Mathematics
University College London
Gower Street
London WC1E 6BT
United Kingdom

March 20, 2002

Abstract

We introduce the braid groups in their connection to knot theory and investigate several of their properties. Based on term rewriting systems, which we review, we find new solutions to the word and conjugacy problems in the braid groups. A similar problem asks for the minimal length word for an equivalence class in a given braid group which we prove to be NP-complete (after a review of this concept) and present a new algorithm for it. As this algorithm takes an exponentially increasing amount of time, we construct an algebraic approximation algorithm which we find to work well. We consider several methods of approximating the minimal word via computer simulation of the braid strings moving under the influence of certain forces. Using the theory of tangles which we also review, we construct a new notation for knots which is usable by a computer. From this notation, we construct an efficient algorithm to find the braid or plat whose closure is ambient isotopic to any given knot. Finally, we apply the computer software developed for these problems to the solar coronal heating problem by simulating magnetic flux tubes. We also present a number of incidental results that were found along the way of researching these problems.

This is dedicated to all those who, in the face of adversity,
throw themselves headlong into the battlefields
and fight until they achieve victory or die trying.

"It is time I focused on my problem. Who does not have a problem? — Everybody has one, and indeed several. Each problem has its rank; the main problem moves to the center of one's life, displacing the other problems. It incessantly haunts us like a shadow, casting gloom on our minds. It is present even when we awaken at night; it pounces on us like an animal. ... When I stir my morning coffee and watch the swirling of the streaks, I am observing the law that moves the universe — in the whirling of the spiral nebulae, in the eddying of the galaxies. ... But what does it matter? Whether the universe whirls or crumbles — the problem remains behind it. ... The problem is indivisible; man is alone. Ultimately, one cannot rely on society. Although society usually wreaks harm, indeed often havoc, it can also help, although not more than a good physician — up to the inevitable limit where his skill fails. ... My time is limited; but anyone can spend a month retreating into the forest or the desert. There, he can describe — or better: circumscribe his problem; it is then defined, though not solved."

Ernst Jünger in *Aladdin's Problem.*

Contents

Abstract **2**

Notation **8**

Acknowledgements **9**

Preface **10**

1 Introduction to Braid and Knot Theory **11**
1.1 Knot Theory . . . 11
1.1.1 The Knot Classification Problem . . . 12
1.1.2 Topological Invariants . . . 13
1.1.3 The Complement of the Knot . . . 14
1.1.4 Peripheral Group System . . . 14
1.2 Knot Notations . . . 16
1.3 Braid Theory . . . 16
1.3.1 The Braid Group B_n . . . 17
1.3.2 Markov's Theorem . . . 20
1.3.3 Stabilization is Non-trivial . . . 20
1.3.4 Strategies for the Markov Problem . . . 21

2 The Word, Conjugacy and Markov Problems **22**
2.1 Properties of the Center of B_n . . . 22
2.1.1 Algebraic Properties . . . 22
2.1.2 Word Problems . . . 24
2.1.3 The Peripheral Group System . . . 25
2.2 A Review of Term Rewriting Systems . . . 27
2.2.1 Word Problems . . . 28
2.2.2 Termination . . . 29
2.2.3 Confluence . . . 29
2.2.4 Completeness . . . 29
2.3 Word and Conjugacy in B_n . . . 30
2.3.1 The Word Problem in B_n . . . 30
2.3.2 The General Conjugacy Problem . . . 33
2.3.3 The Conjugacy Problem in B_n . . . 38

3 The Minimum Word Problem **41**
3.1 Introduction . . . 41
3.2 NP-Completeness . . . 42
3.2.1 A Review of NP-Completeness . . . 42
3.2.2 Practicality of the Theory of NP-Completeness . . . 43
3.2.3 Sorting Does Not Minimally Partition . . . 44
3.3 Non-Minimal Braids is NP-Complete . . . 44
3.3.1 Statement of the Problem . . . 44
3.3.2 The Weft Braids . . . 45
3.3.3 Minimal Weft Braids . . . 47

3.4 Minimal Length Words . . . 47
3.4.1 Minimal Braids Without Increasing Length . . . 48
3.4.2 The Diagram of a Braid . . . 48
3.4.3 Counterexamples . . . 50
3.5 The Size of Diagrams . . . 51

4 Minimal Words via Elastic Relaxation 52
4.1 Introduction . . . 52
4.2 Obtaining and Embedding Random Braids . . . 53
4.2.1 Randomly Generating Algebraic Braids . . . 53
4.2.2 Embedding Algebraic Braids . . . 54
4.2.3 Extracting Geometric Braids . . . 55
4.3 A Crossing Number Minimizing Force . . . 55
4.3.1 Expressions for crossing number . . . 55
4.3.2 Derivation of the crossing number force . . . 56
4.3.3 Simulation Considerations . . . 57
4.4 Energy relaxation . . . 57
4.4.1 The Constrained Elastic Force . . . 58
4.4.2 The Curvature Elastic Energy . . . 58
4.5 Algebraic Minimization . . . 59
4.6 Some numerical results . . . 61
4.6.1 Efficacy Analysis . . . 61
4.6.2 Efficiency Analysis . . . 62
4.6.3 Energy Analysis . . . 63
4.7 Conclusions . . . 65

5 Knotation and Braiding a Knot 67
5.1 Tangles . . . 67
5.1.1 Definition and Partition . . . 67
5.1.2 Classification of Tangles . . . 69
5.2 Knot Notation . . . 69
5.2.1 The Universal Polyhedron . . . 70
5.2.2 Basic properties . . . 73
5.3 Braiding a Knot . . . 75
5.3.1 An Example . . . 75
5.3.2 Platting a Knot . . . 76
5.3.3 Laying the Axis . . . 77
5.3.4 Getting the Braid . . . 79
5.4 Translation from Conway's Knotation . . . 80

6 The Solar Heating Problem 82
6.1 Introduction . . . 82
6.2 The Temperate Photosphere . . . 85
6.3 Generating Arches from Footpoints . . . 86
6.4 The Simulation . . . 86
6.5 Mutual and Self Helicity . . . 87
6.6 The Moving Footpoints . . . 88
6.7 Conclusions . . . 88

List of Figures

1.1 The unknot, Hopf link and Whitehead link . . . 12
1.2 The Reidemeister moves . . . 12
1.3 The 3-simplex . . . 14
1.4 Longitude and meridian . . . 15
1.5 Two forms of double points . . . 15
1.6 Canonical closure of a braid . . . 16
1.7 Plait closure of a braid . . . 17
1.8 The generators of the braid groups . . . 18
1.9 The Markov moves . . . 20

2.1 The braid Δ_3 labeled . . . 25
2.2 The braid Δ_3^2 labeled . . . 27
2.3 The rules for the word problem . . . 31
2.4 Cyclic words illustrated . . . 35
2.5 Proof of Newman's Lemma . . . 36
2.6 Proof of cyclic critical pair lemma . . . 37

4.1 Reduction ratios . . . 62
4.2 Example reductions of all methods . . . 64
4.3 Energy comparison of reduction methods . . . 65

5.1 3-ball . . . 67
5.2 Elementary tangles . . . 68
5.3 Tangle addition . . . 68
5.4 The universal polyhedron . . . 70
5.5 The trefoil knot labeled . . . 71
5.6 The trefoil knot in $P(2,2)$ . . . 72
5.7 Knot addition . . . 73
5.8 An axis for the trefoil knot . . . 75
5.9 The trefoil knot around its axis . . . 76
5.10 The braid of the trefoil knot . . . 77
5.11 The conversion of a knot into a plait . . . 78
5.12 The braiding axis . . . 79

6.1 The solar corona . . . 82
6.2 Flux tubes in the corona . . . 83
6.3 Loop prominence . . . 84
6.4 Schematic flux tube model . . . 85
6.5 The computer model . . . 86
6.6 The helicity angles defined . . . 87
6.7 Photograph of axial twist . . . 88

List of Tables

2.1 Overlaps between rules in the word problem solution 32
2.2 Cyclic overlaps between rules in $\mathcal{W}_n$. 39

3.1 The Size of Diagrams of Fundamental Words 51

4.1 Reduction ratios $\alpha = C_{min}/C(0)$ as a function of the number of strings n. 61
4.2 Total length per string. 64

Notation

The most important symbols used throughout the thesis are explained in the table below.

Symbol	Meaning
$[A, B]$	$AB - BA = 0$
σ_i	generator of braid group, single braid crossing
B_n	braid group of n strings
$\approx$	equivalence in a group
$\approx_C$	conjugate in a group
$\approx_M$	Markov equivalence in braid group
$L(A)$	number of Artin generators in braid A; the length of braid A
$\overline{A}$	(canonical) closure of braid A
$\pi_1(K)$	fundamental group of complement of knot K
$p(K)$	peripheral group system of complement of knot K
$a_{i,j} = \sigma_i\sigma_{i+1}\cdots\sigma_j$ for $i \leq j$	ascending braid word
$d_{i,j} = \sigma_i\sigma_{i-1}\cdots\sigma_j$ for $j \leq i$	descending braid word
$\Delta_n = a_{1,n-1}a_{1,n-2}\cdots a_{1,1}$	fundamental braid, see Δ_n^2
Δ_n^2	generator of the centre of B_n, see Δ_n

Acknowledgements

Body and soul must be kept together while doing mathematics. My gratitude goes to the Graduate School of University College London which has kindly provided this paramount service through awarding me a research scholarship. I am indebted to Mitchell A. Berger, my supervisor, for taking me on, supporting me through my learning and researching, and reading the drafts of papers and thesis while making many helpful comments and suggestions that lead to new and exiting things.

Thanks go to Margaret Yoder for providing insights into her thesis and into rewriting systems, to Claude Marche and Benjamin Monate for all their help, explanations and patience and for the provision of their excellent term rewriting software CiME with which many ideas were tested during the development of the rewriting systems, to Mike S. Paterson for explanations of his work with Alexander A. Razborov and John M. Talbot for excellent explanations of complexity theory. Many other mathematicians helped me during the development of this work and the associated software program BraidLink.

And finally, moral support is vital to such a lonely undertaking. My warmest thanks go to my parents, family and friends who have supported and encouraged me over the years.

Preface

This thesis considers several problems in the theory of braids. Braid theory is a branch of knot theory which is contained within topology. During the discussion of the research, we make use of braid, knot, group and tangle theories as well as techniques from term rewriting systems and NP-completeness which come from computer science, and topology. Other than a basic knowledge of topology and group theory, no further knowledge of any other branch of mathematics or computer science is necessary in order to read this thesis as we review all these fields to the extent necessary for our purposes.

We begin in chapter 1 by reviewing braid and knot theory. In section 2.1, we use group theory and algebra to deduce certain properties of the braid groups. We proceed in section 2.2 to review term rewriting systems and use them in section 2.3 to solve the word and conjugacy problems in the braid groups.

We state the minimum word problem for braids in chapter 3. In section 3.2, we review the theory of NP-completeness and use it to show that the minimum word problem is NP-complete in section 3.3. We find, in section 3.4, an algorithm to solve the problem which runs in exponential time.

The minimum word problem may be approached by simulating braids as elastic strings. This approach works well in practice. In section 4.2, we discuss how to generate a random braid, embed it in space and retrieve a braid word from a set of strings. In sections 4.3 and 4.4, we present three forces that we will use in the simulation. Section 4.5 presents an efficient algebraic heuristic algorithm to solve the problem. The properties of the forces and data to compare them in both efficacy and efficiency is in section 4.6. The simulation contained in chapter 4 was published in a slightly different form in [10].

We review tangle theory in section 5.1. Section 5.2 presents a new notation for knots and gives a few basic properties of it. We solve the problem of turning a knot into a braid or plat in section 5.3 and give translation algorithms between our new notation and existing computer notations in section 5.4.

Apart from the crossing number minimizing force in section 4.3, which is the work of Prof. M A Berger, and the curvature elastic force in section 4.4.2, which is the work of Dr. R Prandi, the contents of the thesis are the author's work except where explicitly cited in the references. For reasons of space we provide the proof of a result only when it or the result is new and refer the reader to the literature if the proof exists therein.

In the investigations described in the above chapters, computer assistance was frequently necessary and for this purpose a program called BraidLink was written in C++ for Microsoft Windows. Many of the algorithms in this thesis are implemented in BraidLink but the functionality of BraidLink goes far beyond them. The program may be obtained from the author, for further information and the manual see `http://www.knot-theory.org`.

For each entry in the bibliography, we provide a list of page numbers on which that particular work was cited. After the bibliography, we provide an index to the technical terms used in the thesis. Page numbers in bold indicate that the term is defined on that page whereas a normal page number simply means that the term is used in an important way on that page.

Chapter 1

Introduction to Braid and Knot Theory

1.1 Knot Theory

Everyone has encountered knots. We use knots to tie our shoelaces, fasten our washing lines and secure ourselves from falling during climbing. Knot theory studies the topology of knots such as these with the only additional requirement that after they are tied, the ends must be glued together never again to be undone. The inherent freedom of topology means that we are allowed to do anything to the knot - stretch, bend, twist and distort it in any way - except cut or glue the string at any point. The modern view of thinking of a knot as a tied piece of string with connected ends is much simpler than the original conception:

"By a knot of n crossings, I understand a reticulation of any number of meshes of two or more edges, whose summits, all tessaraces ($\alpha\kappa\eta$), are each a single crossing, as when you cross your forefingers straight or slightly curved, so as not to link them, and such meshes that every thread is either seen, when the projection of the knot with its n crossings and no more is drawn in double lines, or conceived by the reader of its course when drawn in single line, to pass alternately under and over the threads to which it comes at successive crossings." [93]

The historical roots of knot theory begin in the middle of the nineteenth century when Lord Kelvin (at that time still William Thompson) had the idea that an atomic theory could be created on the basis of vortex knots in the (then accepted) luminiferous ether. The fluid-like ether was thought to be the all pervading medium in which light travels. Different elements of matter were thought to correspond to topologically distinct knots in this model. Thompson asked his friend Peter Guthrie Tait to study knots and to draw up a list of topologically distinct knots. This was the impetus for Tait to create knot theory. When Kelvin approached Tait about constructing a knot table, they envisioned a research programme which was to start by classifying knots (in form of a table), mapping this table to the spectrum of observed elements via further experiments and finally to produce a theory of everything. The 1887 experiment of Michelson and Morley showed that the luminiferous ether does not exist and thus the vortex atom theory was abandoned. However knot theory continued as a mathematical discipline. Tait was primarily concerned with creating a table of topologically distinct knots in order of increasing complexity. The measure of complexity used was the minimum number of crossings over all two dimensional projections of the knot. Tait succeeded in creating a remarkably accurate table of prime knots up to and including ten crossings.

Figure 1.1 gives three examples of knots. The leftmost knot is called the unknot and was originally not regarded as a knot at all. The unknot is a very special case and arguably the most important single knot. The other two are the Hopf and Whitehead links respectively which have two closed loops of string each. The term *link* is usually reserved for a knot with more than one component. The arrows on the diagrams supply an orientation to the knot. The orientation is important because there exist knots for which altering the orientation can change the topology. Many times however, no orientation is specified.

The modern definition of a *knot* K is an embedding of n copies of S^1 into S^3, the three-sphere (thus we use the term *knot* as inclusive of links). Whenever a new mathematical object is defined,

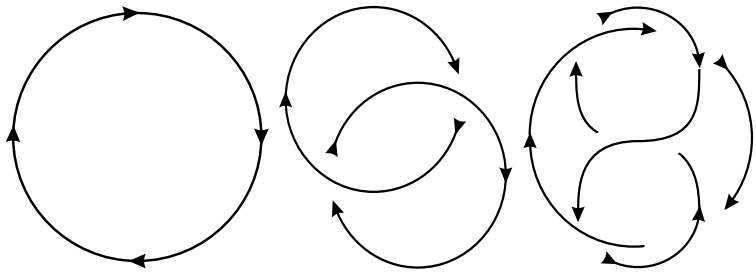


Figure 1.1: The unknot, Hopf link and the Whitehead link. These knots are oriented as indicated by the arrows. Knots do not have to be oriented but every knot is orientable and different orientations may not be deformable into each other without cutting or gluing, i.e. they may be topologically distinct.

the question arises how equality is to be defined. In knot theory this is far from obvious and there were several contending views very early on. Tait [142] viewed knots based on string and allowed axial twisting of the rope while Kirkman [93] viewed knots based on ribbons and did not allow axial twists. This lead to different tables of distinct knot types and created some confusion.

The definition of equivalence is based on the topological concept of ambient isotopy and is essentially the same definition that Tait used. Two embeddings $k_1, k_2 : X \to Y$ are *ambient isotopic*, denoted by $\approx$, if there is a level preserving isotopy H such that

$$H : Y \times I \to Y \times I, H(y,t) = (h_t(y), t) \tag{1.1}$$

where $k_2 = h_1 k_1$, $h_0 = id_Y$ and $I = [0,1]$. H is called the *ambient isotopy*. This means that if we can take one knot and distort into another smoothly without any discontinuities, then they are both the same knot. If we must go through some discontinuities, i.e. if we must cut or glue, then the two are not the same knot. From this definition, it is clear that the unknot and the Hopf link in figure 1.1 are not the same knot. The reason is that the Hopf link has two components which could be reached from the unknot's single component only by cutting and gluing. This process of cutting and gluing is commonly referred to as *surgery*.

Having defined what it means for a knot to be equal to another, we ask for a method to discover if the equality holds between any two given knots. This is the classification problem for knot theory and no satisfactory answer has yet been given.

1.1.1 The Knot Classification Problem

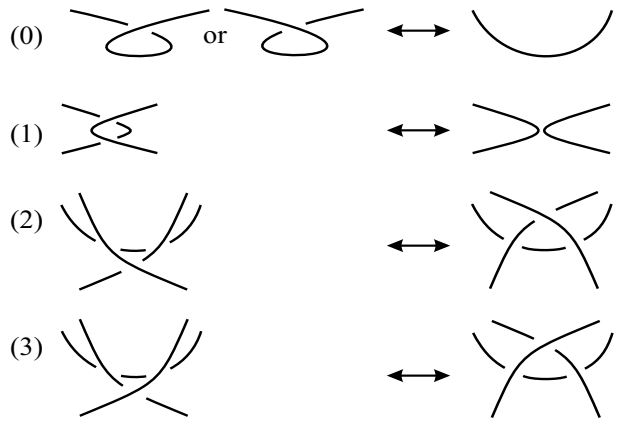


Figure 1.2: The Reidemeister moves.

The overriding problem in constructing a knot table is the difficulty of determining whether two knot diagrams are topologically distinct or not. It is possible to construct all possible knot

diagrams up to a given number of crossings using an (essentially) algebraic method [143] but distinguishing these is the real problem. This enumeration method has been refined [58] and used to tabulate knots based on topological invariants (see below) up to and including 17 crossings [69] (for an excellent review on the history of tabulation and how it is done using a computer see [147]). Therefore a practical method for comparing two given knot diagrams would make it possible to construct a complete knot table up to a certain number of crossings, i.e. such a method would classify knots. After Tait, Reidemeister [130] showed that two knot diagrams are equivalent if and only if they can be transformed into each other via a set of four moves which are called Reidemeister moves, see figure 1.2. While this turns the problem into a combinatorial one, it is often necessary to further complicate a diagram in order to fully simplify it later. Making this transformation is not readily amenable to algorithmic manipulation. Thus Reidemeister's moves do not present a practical method to distinguish knots. They make it easy however, to prove the invariance of other properties of knots. If one can show that a particular function $f(K)$ calculated from a knot K is invariant under all the Reidemeister moves, then $f(K)$ is a topological invariant of knots. This means that if $K_1 \approx K_2$, then $f(K_1) = f(K_2)$. Many such functions have been found but it can be shown that for most known functions it does not follow that if $f(K_1) = f(K_2)$, then $K_1 \approx K_2$. In that sense, most topological invariants are *incomplete*. An invariant is called *complete* when $K_1 \approx K_2$ if and only if $f(K_1) = f(K_2)$. It is the holy grail of knot theory to find a (readily computable) complete invariant. It is not known whether such an invariant actually exists. As we will discuss below, the complement (R^3 with the knot removed) of the knot is a complete invariant but distinguishing these, while possible, is such a time consuming affair, that this method of classifying knots is not practical [77].

We can define a knot sum $K_1 \# K_2$ between two knots K_1 and K_2 by cutting both knots at one arbitrary point and splicing the ends together in such a way that the orientations, if any, are compatible. It can be shown that this sum is independent of the choice of the points and thus dependent only upon which components of K_1 and K_2 are cut [39]. It can also be shown that there is no inverse to this sum, that is, in general, there is no knot K^{-1} for any knot K such that $K \# K^{-1} \approx U$ where U is the unknot of as many components as K has [39]. Therefore, the knot isotopy problem does not reduce to recognizing the unknot, which is a fundamental complication of the problem. A knot is called *prime* if it can not be represented as the sum of two non-trivial knots, it is called *composite* otherwise.

1.1.2 Topological Invariants

The topological invariants of knots fall into a number of categories. A trivial invariant is the number of components but since there are a large number of obviously distinct knots for each value, this is not a very strong invariant even though it is easily computable. Amongst the simplest to state are the invariants which are defined as the minimum of quantities over all possible diagrams of a knot. Since there are an infinite number of diagrams for each knot, these invariants are difficult to determine and for many of them, there exists no general method. Examples of this are the minimum crossing number, bridge number and braid index [116]. Another important category is formed by the polynomial invariants. A number of polynomials have been defined which are topological invariants of a knot. The polynomial is generally calculated via a topological form of recursion relation, called a "skein relation," which we will not go into [91]. The Jones polynomial and its generalization, the Homfly polynomial, are very important in several applications of knot theory as well as knot theory itself. They are very powerful invariants but there are still an infinite number of knots with identical polynomials. The fundamental point to note is that, while polynomial invariants are among the most powerful knot invariants, the amount of computing time required to determine them increases exponentially with the number of crossings in the diagram of the knot. For a review on knot polynomials, see [99].

The fact that it is unknown whether there exist non-trivial knots for which the Jones polynomial is equal to one (the value for the unknot), shows that these invariants are not fully understood at present. Recognizing the unknot is a subproblem of the knot classification problem and if the above question is negatively resolved, then the Jones polynomial would provide the best known unknot detection mechanism (from a computational point of view) [21]. There are two algorithms which can distinguish the unknot: One due to Haken [74] [75] which was the precursor to his classification of 3-manifolds and one due to Birman and Hirsch [21] which makes use of closed braids. While the

former is clearly exponential in execution time, the later has not been analyzed for complexity but appears to be exponential. It has not been analyzed what the complexity of the knot classification problem in general is but it would seem to be easier to distinguish a knot from the unknot than to distinguish two arbitrary knots.

1.1.3 The Complement of the Knot

We define a *knot* K as an embedding of n copies of S^1 into S^3, the three-sphere. Consider the knot K and surround it with a tubular neighborhood $V(K)$, then the manifold $C(K) = \overline{R^3 - V(K)}$ will be called the *complement* of K. It can be shown that for any knots K_1 and K_2, $K_1 \approx K_2$ if and only if there exists orientation preserving homeomorphism $H : C(K_1) \rightarrow C(K_2)$ [39]. Thus the knot complement is a complete invariant of the knot.

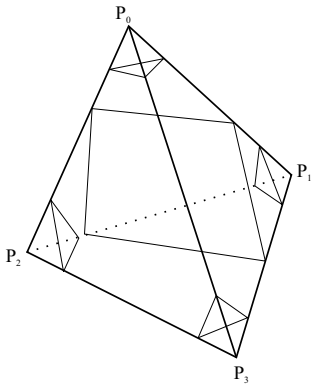


Figure 1.3: The standard 3-simplex or tetrahedron.

$C(K)$ is clearly a 3-manifold and it can thus be distinguished (or otherwise) from other 3-manifolds, in particular other knot complements $C(K')$, by Haken's classification of 3-manifolds. We briefly present the idea of the method but refer the reader to [77] for a pedagogical treatment. First, a triangulation must be found on R^3. A triangulation for a 3-manifold essentially consists of filling the manifold with non-overlapping tetrahedra in such a way that any point in the manifold is in a tetrahedron, see figure 1.3. Any surface in the manifold will now intersect some tetrahedra. These intersections will be triangles (2-simplices) or squares, see figure 1.3. Since the tetrahedra do not overlap but fill all of the manifold, the number of intersections of a surface with adjacent sides of tetrahedra must be equal. This requirement gives a set of equations describing the surface in the manifold. Since the triangulation is not unique, neither is the set of equations. Comparing two knot complements has been a topological problem but this construction turned it into a combinatorial one. If we can compare the set of equations from two surfaces (thickened knot neighborhoods of K and K'), then we can distinguish the knots. This can be done [77] but the time taken is exponential in the number of crossings of the knot, so exponential that the algorithm can not be used to practically distinguish knots even of small crossing number.

1.1.4 Peripheral Group System

The complement of a knot is uniquely specified (up to isomorphism) by its peripheral group system which consists of the fundamental group and a few subgroups thereof (this is Waldhausen's theorem [153], see [78] for a more accessible proof). This is the only complete invariant which it is practical to actually calculate but since group isomorphism is algorithmically untestable (the Adian-Rabin theorem [2] [3] [129]), this does not provide a practical method to distinguish knots either. It is however known that the word problem for any fundamental group of any knot is solvable [154]. If the knot is alternating, the conjugacy problem is also solvable [5].

We define the *linking number* of two curves a and b, denoted by $lk(a, b)$ as the weighted sum of the characteristics ϵ of each crossing. The characteristic ϵ is -1 or 1 depending on whether the crossing matches respectively with the first or second of the two possible scenarios for a crossing

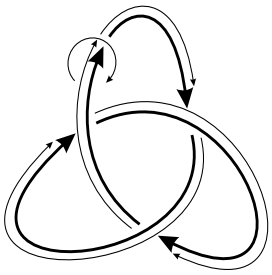


Figure 1.4: The thick curve displays the trefoil knot with an orientation. The thin curve which is parallel to the trefoil knot is the longitude; the orientation of the longitude is the same as the knot. The thin curve encircling both trefoil and longitude at the top left hand corner is the meridian. Note that the five conditions given in the text are fulfilled by these curves.

shown in figure 1.5. We define a *meridian* m_i and a *longitude* l_i of a knot component K_i by requiring the following properties: (1) m_i and l_i are oriented, polygonal, simple and closed curves in $\partial V(K_i)$, the boundary of the thickened neighborhood of K_i which we denote by $V(K_i)$, (2) m_i and l_i intersect in exactly one point, (3) m_i is null homologous ($m_i \sim 0$) in $V(K_i)$ and $l_i \sim K_i$ in $V(K_i)$, (4) $l_i \sim 0$ in $C(K_i)$ and (5) $lk(m_i, K_i) = 1$ and $lk(l_i, K_i) = 0$ in S^3. The above five properties define m_i and l_i uniquely up to isotopy on the boundary of $V(K)$ [39] (see figure 1.4 for an illustration). The *meridian-longitude system pair* $\mathcal{M}(K)$ for a j-component knot K is the pair of sets $(\{m_1, m_2, \cdots, m_j\}, \{l_1, l_2, \cdots, l_j\})$.

The *knot group* $\pi(K)$ is the fundamental group of $C(K)$, $\pi_1(C(K), b)$ where b denotes a base point. The meridians and longitudes of a meridian-longitude system pair $\mathcal{M}(K)$ of the knot K may be considered to be elements of $\pi(K)$ by choosing a path p_i in $C(K)$ from the base point b to the (unique by definition) point $m_i \cap l_i$ for each i. Then the subgroup $\langle m_i, l_i \rangle$ of $\pi(K)$, generated by m_i and l_i is independent of the choice of p_i up to conjugation. The *peripheral group system* of a j-component knot K is $p(K) = (\pi(K); \mathcal{M}(K))$. By an isomorphism ϕ between two peripheral group systems $p(K) \approx_\phi p(K')$, we mean $\pi(K) \approx \pi(K')$ such that $\phi(m_i) = m'_i$ and $\phi(l_i) = l'_i$ for all i. It can be shown that for any two knots K_1 and K_2, $p(K_1) \approx p(K_2)$ if and only if $K_1 \approx K_2$ [92]. If we restrict attention to prime knots of a single component, we have $\pi(K_1) \approx \pi(K_2)$ if and only if $K_1 \approx K_2$ [92]. Thus the problem of knot isotopy can be transformed into the problem of peripheral group system isomorphism. Since it is not possible to determine, in general, if two groups are isomorphic, this does not solve the knot classification problem.

Figure 1.5: The two possible forms of double points in the diagram of an oriented knot.

There exists a simple method due to Wirtinger, to find a presentation of $\pi(K)$ from a diagram of K. Suppose there are n arcs in the diagram. We label the i^{th} arc by x_i. The set $\{x_i\}$ for $1 \leq i \leq n$ generates $\pi(K)$. Every crossing in the diagram of K is of one of the two kinds displayed in figure 1.5. For each crossing determine its type and add the relation $x_l x_i x_k^{-1} x_i^{-1} \approx e$ or $x_l x_i^{-1} x_k^{-1} x_i \approx e$ to the group respectively. The resulting group is $\pi(K)$ defined by its *Wirtinger presentation*. It is a practical observation that this presentation can often be simplified considerably in that some generators are removable [65]. In particular, $\pi(K)$ for the torus knot $T_{p,q}$, which is a knot that winds around a torus p times the short way around (meridionally) and q times the long way around (longitudinally), is given by $\pi(K) = \langle \{a, b\} : a^p = b^q \rangle$ [92].

Even though $\pi(K)$ can not be readily used as a practical invariant, it is however a convienient starting point to define many other invariants, for example the Alexander polynomial [148] which

was the first of the polynomial invariants and (like the Jones polynomial) revolutionized knot theory.

1.2 Knot Notations

Knot theory has gained tremendous momentum from proofs that certain mathematical objects are isotopy invariants of knots such as the peripheral group system discussed above. Such proofs and general statements about knots form a large part of knot theory but in applications of knot theory, actual computation of these objects is often necessary. Therefore, it is important to have a practical method of computation for such invariants. Some invariants, such as the unknotting number, can not yet be calculated in an algorithmic manner for every knot. Other invariants can only be calculated by algorithms whose complexity increases exponentially, thus rendering them useless for all but small knots. There exist only a few invariants which may be calculated easily.

Because it is so laborious to compute many interesting properties of a particular knot, the use of computers is essential. However if a computer is to be used, the search for an efficient algorithm becomes important. The pivot of all algorithms is the form of the input. For many physics calculations, for example, the choice of coordinate system often allows far greater simplification of the calculations than a change in computational procedure. Therefore, while the algorithm is important, a good notation for knots is paramount. Currently there are several different systems of "knotation" (the term was coined by John Conway in a popular lecture with this title) which are widely used, we shall illustrate two of them: Conway's [50] and Dowker and Thistlethwaite's [58].

Conway's knotation relies on setting up templates for knots which he calls basic polyhedra. One inserts standard knot pieces called tangles into the vertices of the template (tangles are introduced in chapter 5). This knotation is quite intuitive since the geometrical aspects of the knot projection can be immediately visualized it is however non-trivial to construct the notation given a knot projection and the notation is limited to knots with few crossings without making necessary extensions.

The Dowker-Thistlethwaite code for a knot is an improvement of Tait's notation. One chooses a point on the knot at random and follows it in the direction of its orientation. The crossings are named "A", "B", "C" and so on in the order that they are met and one writes down which ones one meets in order. It can be shown that only every second letter is needed as the others can be recovered and so the notation for a knot projection of n crossings contains n letters. Implemented algorithms to calculate most invariants from this code exist. The main application of this code is in the computer-assisted tabulation of knots [147]. In chapter 5, we introduce a new notation which will allow us to transform a knot into a closed braid (see next section).

1.3 Braid Theory

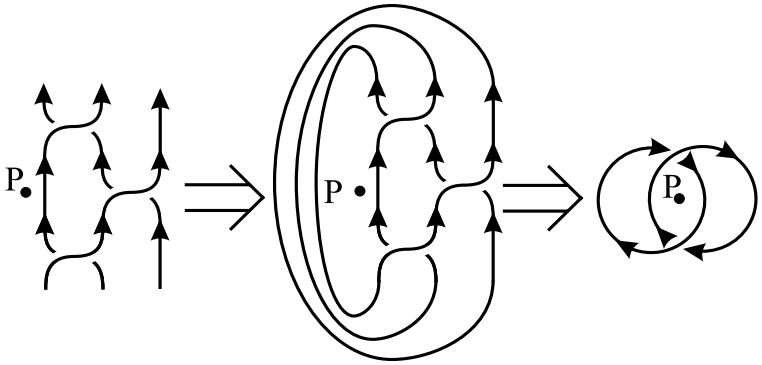


Figure 1.6: The (canonical) closure of a braid. In the braid group language, the braid is $\Delta_3 = \sigma_1\sigma_2\sigma_1$ and the knot is the Hopf link.

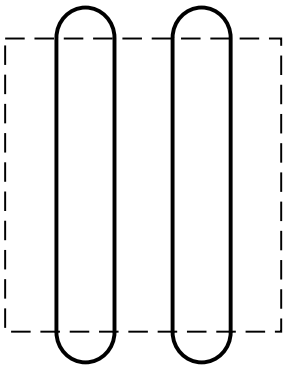


Figure 1.7: The plait closure of a braid. Note that there is potential conflict between orientations of the braid strings in the plait closure; it becomes impossible to plait a braid in which all strings are oriented in the same way.

In 1923, Alexander proved that any knot projection can be modified via Reidemeister moves into a form with respect to a special point P in the plane which has the property that for a point A which traverses the knot in the direction of its orientation, a plane perpendicular to that of the projection intersecting both P and A rotates around P in a constant direction (clockwise or anti-clockwise but never both) [4]. When Artin invented braids [7], it was noticed that if one specified a point outside the braid to be P and connected the top and bottom ends of the braid's strings with each other in such a way that the connecting lines circumnavigated P (this process is called *closing* a braid, illustrated in figure 1.6) and oriented the braid's strings in a uniform (upwards or downwards) direction, one had obtained exactly this form. Thus Alexander had shown that every (oriented) knot can be represented by a closed (oriented) braid. In his paper, Artin had found a group structure for braids which defined open braid isotopy - that is topological equivalence of two braids under the restriction that the endpoints remain fixed. The method of closing a braid which is illustrated in figure 1.6 is called the canonical closure to distinguish it from the plait closure. In the plait closure, we join neighboring ends together as illustrated in figure 1.7. It is necessary for the braid to have an even number of strings for the plait closure.

1.3.1 The Braid Group B_n

The braid group B_n for a braid of n strings is generated by single crossings. Suppose that all strings are vertical apart from strings i and $i+1$ which cross over each other. If i overcrosses $i+1$, we denote this by σ_i and the inverse is denoted by σ_i^{-1} (see figure 1.8 for an illustration). The set $\{\sigma_1, \sigma_2, \cdots, \sigma_{n-1}\}$ generates the group B_n and together with their inverses $\{\sigma_1^{-1}, \sigma_2^{-1}, \cdots, \sigma_{n-1}^{-1}\}$, B_n satisfies the defining relations

$$\sigma_i \sigma_i^{-1} \approx e \tag{1.2}$$

$$\sigma_i \sigma_j \approx \sigma_j \sigma_i \quad \text{for} \quad |i-j| > 1 \tag{1.3}$$

$$\sigma_i \sigma_{i+1} \sigma_i \approx \sigma_{i+1} \sigma_i \sigma_{i+1} \tag{1.4}$$

where e denotes the identity element. Topologically e is the braid of n strings without any crossings; i.e. n vertical strings. The generators σ_i are called *Artin generators*. The proof that the topological equivalence relation of braids is identical to the group theoretical equivalence relation defined by the equations above under the map that a crossing in the topological braid is interpreted as a generator in the group is given in [8] or more accessibly in [39]. We shall use $\approx$ to denote equivalence under a given set of identities and $=$ to denote exact (letter by letter) equivalence.

We will call a word *positive* if it contains only generators and no inverses; the inverse of a positive word is called *negative*. We shall call two positive braids A and B *positively equal* if there exists a sequence of braids W_i for $0 \leq i \leq q$ with $W_0 = A$, $W_q = B$, W_j and W_{j+1} different by a single application of the braid group's defining relations and W_i all positive. Garside has shown that if two positive braids are equal, they are positively equal [64].

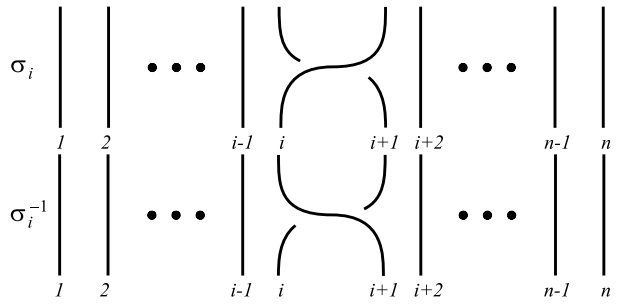


Figure 1.8: The generator σ_i and its inverse σ_i^{-1} for the braid group B_n.

The length of braid A in terms of Artin generators will be denoted by $L(A)$. A general braid $A \in B_n$ may be written in the form

$$A = \sigma_{i_1}^{j_1} \sigma_{i_2}^{j_2} \cdots \sigma_{i_{L(A)}}^{j_{L(A)}} \tag{1.5}$$

where $1 \leq i_k < n$ and $j_k = \pm 1$ for any $k : 1 \leq k \leq L(A)$. We define the *exponent sum*, denoted $\exp(A)$ of A by

$$\exp(A) = \sum_{k=1}^{L(A)} j_k \tag{1.6}$$

It can be shown that $\exp(A)$ is a conjugacy class invariant (and hence equivalence class invariant) of A [20]. If A is as in equation (1.5), then we define the *reverse operator* R by

$$R(A) = \sigma_{i_{L(A)}}^{j_{L(A)}} \sigma_{i_{L(A)-1}}^{j_{L(A)-1}} \cdots \sigma_{i_1}^{j_1} \tag{1.7}$$

and call $R(A)$, the *reverse* of A.

We define three special braid words: The *fundamental* braid Δ_n, the *ascending* braid $a_{i,j}$ and the *descending* braid $d_{i,j}$ by

$$\begin{aligned}
a_{i,j} &= \sigma_i \sigma_{i+1} \cdots \sigma_j \qquad i \leq j && (1.8)\\
d_{i,j} &= \sigma_i \sigma_{i-1} \cdots \sigma_j \qquad j \leq i && (1.9)\\
\Delta_n &= \sigma_1 \sigma_2 \cdots \sigma_{n-1} \sigma_1 \sigma_2 \cdots \sigma_{n-2} \cdots \sigma_1 \sigma_2 \sigma_1 && (1.10)\\
&= a_{1,n-1} a_{1,n-2} \cdots a_{1,2} a_{1,1} && (1.11)
\end{aligned}$$

Δ_n is important in braid theory because Δ_n^2 generates the center of the braid group B_n [43]. Garside [64] has shown that the fundamental braid satisfies

$$\begin{aligned}
\Delta_n \sigma_i &\approx \sigma_{n-i} \Delta_n = \widehat{\sigma_i} \Delta_n && (1.12)\\
R(\Delta_n) &\approx \Delta_n && (1.13)
\end{aligned}$$

where we have defined $\widehat{\sigma_i} = \sigma_{n-i}$. We now prove a crucial proposition.

Proposition 1.3.1 *For any σ_i^{-1}, we have $\sigma_i^{-1} \approx \Delta_n^{-1} \Delta_{n-1} d_{n-1,i+1} d_{i-1,1}$.*

Proof. Since $\Delta_n = a_{1,n-1} a_{1,n-2} \cdots a_{1,2} a_{1,1}$ and $R(\Delta_n) \approx \Delta_n$, we have $\Delta_n^{-1} \approx d_{n-1,1}^{-1} d_{n-2,1}^{-1} \cdots d_{1,1}^{-1}$. Thus $d_{n-1,1}^{-1} \approx \Delta_n^{-1} d_{1,1} d_{2,1} \cdots d_{n-2,1}$. From the definition of $d_{i,j}$, we have

$$\begin{aligned}
\sigma_i^{-1} &\approx \sigma_{i-1} \sigma_{i-2} \cdots \sigma_1 d_{n-1,1}^{-1} \sigma_{n-1} \sigma_{n-2} \cdots \sigma_{i+1} && (1.14)\\
&\approx d_{i-1,1} d_{n-1,1}^{-1} d_{n-1,i+1} && (1.15)\\
&\approx d_{i-1,1} \Delta_n^{-1} d_{1,1} d_{2,1} \cdots d_{n-2,1} d_{n-1,i+1} && (1.16)\\
&\approx \Delta_n^{-1} a_{n-i+1,n-1} d_{1,1} d_{2,1} \cdots d_{n-2,1} d_{n-1,i+1} && (1.17)\\
&\approx \Delta_n^{-1} a_{n-i+1,n-1} \Delta_{n-1} d_{n-1,i+1} && (1.18)\\
&\approx \Delta_n^{-1} \Delta_{n-1} d_{i-1,1} d_{n-1,i+1} && (1.19)\\
&\approx \Delta_n^{-1} \Delta_{n-1} d_{n-1,i+1} d_{i-1,1} && (1.20)
\end{aligned}$$

for any i, which proves the proposition. □

Given two words $\alpha, \beta \in B_n$, the decision problem of whether $\alpha \approx \beta$ is called the *word problem*. The word problem in the braid groups was first solved by Artin [8] and therefore provided a solution to the problem of braid isotopy. For two words $\alpha, \beta \in B_n$, if there exists a word $\gamma \in B_n$ such that $\alpha \approx \gamma\beta\gamma^{-1}$ then α and β are called *conjugate*, which is denoted by $\approx_c$. If $\gamma \approx e$, then $\alpha \approx_c \beta$ implies $\alpha \approx \beta$ and thus the *conjugacy problem*, the existence decision of such a γ, contains the word problem as a special case. The conjugacy problem for B_n was first solved by Garside [64]. The best known algorithm for the word problem was formulated by Birman, Ko and Lee [22] with complexity $O(nL^2)$, where L denotes word length, and for the conjugacy problem by Thurston [59] and Birman, Ko and Lee [22] with exponential complexity.

Construct the set $D(A)$ of all words obtainable from A by rearranging of generators, this is called the *Cayley diagram* of A. This set is constructed recursively from A. The first set is $D_0(A) = \{A\}$ and each set $D_i(A)$ is obtained from $D_{i-1}(A)$ by adding all words which can be obtained from the members of $D_{i-1}(A)$ by a single application of the relations 1.3 and 1.4 and are not already members of the sets $D_j(A)$ for $0 \leq j < i$. It is a theorem of Garside [64] that this construction process terminates in a set $D_k(A)$ for finite k and that thence the set $D(A)$ which is the union of all the $D_i(A)$ for $0 \leq i \leq k$ is finite and readily constructible. Since we do not allow cancelations or introductions of generators by use of the relation 1.2, it is an obvious property of $D(A)$ that all members are of equal length $L(A)$.

For any two braids $A, B \in B_n$ we say that A is *prime* to B if and only if $D(A)$ does *not* contain a word in the form $A \approx A_1 B A_2$. Let the number of inverse generators in a braid A be $s(A)$, then proposition 1.3.1 together with equation (1.12) implies that any braid $A \in B_n$ may be written in the form $A_{max} = \Delta_n^{-s(A)} A'$ where A' is positive; the reason for naming it A_{max} will become apparent later on. We obtain this form by replacing each inverse generator in A by the form given in proposition 1.3.1 and then using equation (1.12) to bring all the fundamental braids to the front.

In his celebrated solution to the word and conjugacy problems in B_n, Garside [64] presents an algorithm to put A' into the form $A' = \Delta_n^q A''$ where A'' is prime to Δ_n and another algorithm to put A'' into a form minimal in lexicographical order on the set of generators for the ordering $\sigma_i < \sigma_j$ if and only if $i < j$, which we call $\underline{A}$. Garside shows that the resulting form $A_G = \Delta_n^{-s(A)+q} \underline{A}$ for the braid A is unique. We call $q - s(A)$ the *Garside exponent* and $\underline{A}$ the *Garside remainder* of the braid A. Garside's original algorithms have exponential complexities in n and $L(A)$, however Jacquemard constructed an algorithm with complexity $O(n^7 L(A)^3)$ [83].

In stating the complexities of all algorithms here, we implicitly assume that n and $L(A)$ are independent. Clearly, we may choose a braid $A \in B_n$ of any length at all and thus it would appear that we are justified in this assumption. In the class of non-splittable braids (A braid $A \in B_{n+m}$ is said to be *splittable* if and only if it may be written in the form $A \approx \alpha O_n(\beta)$ where $\alpha \in B_n$, $\beta \in B_m$ and the operator O_n is defined by $O_n(\sigma_i) = \sigma_{i+n}$ [20]) this is not true as we must have $L(A) \geq n - 1$. As in the case of worst-case complexity measurements, we are interested in the asymptotic behavior as n and $L(A) \to \infty$, we may continue to assume that n and $L(A)$ are independent. Should this in some circumstances turn out to be false, the above argument shows that then n is of the order of $L(A)$.

If a knot K is represented by a closed n-braid β, then the mirror image K^* of K is represented by the closed braid β^{-1} and the reverse $K^\dagger$ (obtained by reversing the orientations) of K is represented by $R(\beta)$ or by $R(\beta)^\dagger$ [117].

We will henceforth represent a commutation relation $AB = BA$ by writing $[A, B]$. Put $\alpha = a_{1,n-1}$ and $\beta = \sigma_1$, then it can be shown that another presentation of the braid group is [51]

$$B_n = \left\langle \{\alpha, \beta\} : \alpha^n = (\alpha\beta)^{n-1}; \left[\beta, \alpha^{-j}\beta\alpha^j\right] \ 2 \leq j \leq \frac{n}{2} \right\rangle \tag{1.21}$$

where α^n is the generator of the center $Z(B_n)$ [43]. In this formulation we trivially find the following helpful simplifications

$$\alpha^{-1} = \alpha^{-n}\beta(\alpha\beta)^{n-2} \tag{1.22}$$

$$\beta^{-1} = \alpha^{-n}(\alpha\beta)^{n-2}\alpha \tag{1.23}$$

$$\alpha^{-1}\beta^{-1} = \alpha^{-n}\beta(\alpha\beta)^{n-3}\alpha \tag{1.24}$$

$$\beta^{-1}\alpha^{-1} = \alpha^{-n}(\alpha\beta)^{n-2} \tag{1.25}$$

1.3.2 Markov's Theorem

Given a knot, we may produce an equivalent knot by taking any segment and twisting it about an axis in the projection plane by π while keeping the rest of the knot stationary. This procedure corresponds to the zeroth Reidemeister move (see figure 1.2) and adds one crossing to the diagram. Any crossing of this type is called *nugatory*. If we represent a knot by a closed braid by virtue of Alexander's theorem, we may also add such nugatory crossings via a combinatorial move, called the *Markov* or *stabilization* move (see figure 1.9). Stabilizing a braid $\alpha \in B_n$ corresponds to the operation $\alpha \rightarrow \alpha\sigma_n^{\pm 1}$ or its inverse. Clearly stabilization increases or decreases the number of strings in the braid and so represents a move in the family of braid groups as opposed to the conjugacy and equivalence moves which are contained in a single braid group.

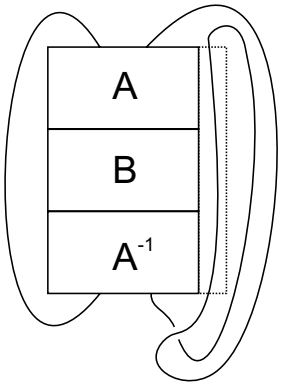


Figure 1.9: Both conjugacy and stabilization are displayed here. We begin with braid B. Conjugation surrounds B with A and A^{-1} on opposite sides which clearly cancel due to the closure. Stabilization introduces a simple loop at the bottom right of the braid, adds a new string to the braid and thus increases the braid group index by one.

Markov stated in 1935 [105] that two closed braids are topologically equivalent if and only if they differ by stabilization and conjugacy moves (recall that conjugacy contains equivalence). This statement became known as Markov's theorem and was first proven in [20]. In its original form, Markov's theorem assumes that the closed braid is embedded in S^3 or R^3, this can however, be generalized to an arbitrary 3-manifold [97]. Markov's theorem transforms the link isotopy problem to a combinatorial question about braids. If two braids $\alpha \in B_n$ and $\beta \in B_m$ (with n and m possibly different) are related by stabilization and conjugacy, they are called *Markov equivalent* which is denoted by $\approx_M$. The decision problem of whether $\alpha \approx_M \beta$ is called the *Markov problem* or the *algebraic link problem*. It is possible to find a single move of which both stabilization and conjugacy are special cases and to formulate, in this way, Markov equivalence in terms of this so called L-move [98]. While this L-move is intuitive, it is not obvious whether the problem has been simplified by this reformulation.

1.3.3 Stabilization is Non-trivial

The first question which arises is whether there exist non-conjugate Markov equivalent braid words in the same braid group, that is whether a solution to the conjugacy problem will solve the Markov problem. This is negatively resolved by showing that the two 4-braids $\alpha = \sigma_1^m\sigma_2^n\sigma_1^p$ and $\beta = \sigma_1^m\sigma_2^p\sigma_1^n$ with m, n, p different, odd and at least three in absolute value are not conjugate but Markov equivalent [118]. It might be thought that it should be possible to reduce the number of strings in a closed braid equivalent of the unknot to one. This is true as all equivalent closed braids can be reached from each other via Markov's theorem but the transition involves, in general, increasing the number of strings before they may be reduced to a single string. In other words, a greedy reduction of strings does not reach the minimum string number, also known as the braid index (not even for the unknot representatives) [114].

It is a practical observation that finding a series of moves to demonstrate the Markov equivalence of two closed braids is very difficult. The difficulty of finding such a sequence has lead Birman

to believe that it may be simpler to solve Markov equivalence for two braids representing prime knots. While this may be true, it is not, in general, easy to decide whether a braid represents a prime knot. Schubert [132] proved that the factorization sequence of a composite knot is unique and has found an algorithm [133] which finds it. This algorithm, consequently, is able to decide whether a knot is prime. However, the execution of the algorithm rests on Hemion's algorithm since it must identify the prime factors of the knot, thus no longer necessitating a solution of the Markov problem since it already solves the link isotopy problem (albeit impractically so). This also shows that this method of deciding primality is not practical. Birman conjectures that a braid represents a prime knot if and only if it is not conjugate to a split braid.

Furthermore, if Birman's conjecture is true and we were to find an algorithm to decide whether a braid was conjugate to a split braid, we would have to solve the Markov problem for this restricted class of braids. If this could be done, we would have a solution to the Markov problem since every braid could be decomposed into its split components and pair-wise tested for non-split Markov equivalence. This would not only resolve isotopy but also give the unique prime knot factorization of the knots. Birman's conjecture is unproven and there exists no algorithm to test whether a braid is conjugate to a split braid. It is possible, however, to solve the Markov problem for certain quotient groups of the braid groups [30].

1.3.4 Strategies for the Markov Problem

Since the word and conjugacy problems are contained in the Markov problem, solutions for these are desirable and have been given numerous times as mentioned before. The stabilization move represents the final hurdle before link isotopy is algorithmically decidable and thus it would be interesting to know when a braid $\alpha \in B_{n+1}$ is conjugate to a braid $\gamma\sigma_n^{\pm 1}$ where γ contains only the generators σ_i for $1 \leq i \leq n-1$, for then one could reduce α to γ using the Markov move. While this has been done [107], the algorithm depends on Garside's conjugacy algorithm [64] which has exponential complexity. Moreover, if two braids were reduced in this way to the minimum string number, they are not, in general, conjugate in this final braid group if they are Markov equivalent and thus this decision procedure does not solve the Markov problem either.

We have defined the exponent sum $exp(\alpha)$ of a braid α as the sum of the exponents of the Artin generators of α. It is obvious that the exponent sum is a conjugacy class invariant but not a Markov class invariant because of stabilization. Thus it is possible for two braids to be Markov equivalent and have different exponent sums. In getting from one braid to the other, the exponent sum must be made equal somewhere in the chain of moves; this can clearly only be accomplished using stabilization. Stabilization can increase or decrease the exponent sum depending whether we add σ_n or σ_n^{-1} or remove either of these. It also changes the number of strings. We may think that starting from a positive braid, we should be able to reach any Markov equivalent positive braid by going through a pure positive sequence of braids; that is, we may think that positive Markov equal braids are positively Markov equal. We note that this would only be possible if the difference in exponent sum between the two braids was precisely their difference in number of strings. We conjecture that positive Markov equal braids are not positively Markov equal.

Much work was done by Birman and Menasco on various properties of links which could be determined from their closed braid representatives (this work was published in the six-paper series [23], [24], [25], [26], [27] and [28]). They prove that there exists a complete numerical invariant for knots but find this invariant only for knots which are closed 3-braids. The invariant for closed 3-braids is described extensively and can be used to determine the braid index and whether the knot is split, composite, amphicheiral or invertible. They also define a new type of move on braids, the exchange move, and prove a Markov-like theorem for it. See [29] for a summary of this work.

Chapter 2

The Word, Conjugacy and Markov Problems

2.1 Properties of the Center of B_n

By the definition of the fundamental word Δ_n, we have that

$$\Delta_n = a_{1,n-1}a_{1,n-2}\cdots a_{1,2}a_{1,1} \tag{2.1}$$

It can be shown [64] that $R(\Delta_n) \approx \Delta_n$. It is clear that $\Delta_{n+1} = a_{1,n}\Delta_n$ and so we have an recursive formula for obtaining the fundamental word of a higher braid group in terms of the fundamental word of a lower braid group. Chow first showed that Δ_n^2 generates the center of B_n. The center plays an important role in what is to follow and we shall have to develop some properties of it; while most are simple to derive, they have nevertheless not been published to the author's knowledge.

2.1.1 Algebraic Properties

Since we have a simple recursion relation for Δ_n, we first prove a similar relation for Δ_n^2.

Proposition 2.1.1 *If* $\zeta_n = R(a_{1,n})\, a_{1,n}$, *then*

$$\Delta_{n+1}^2 \approx \zeta_n\Delta_n^2 \approx \Delta_n^2\zeta_n \tag{2.2}$$

Proof. Recall that $\sigma_i\Delta_n \approx \Delta_n\sigma_{n-i}$. Consider

$$\begin{aligned}
\Delta_{n+1}a_{1,n} &= \Delta_{n+1}\sigma_1\sigma_2\cdots\sigma_n && (2.3)\\
&\approx \sigma_n\sigma_{n-1}\cdots\sigma_1\Delta_{n+1} && (2.4)\\
&\approx R(a_{1,n})\Delta_{n+1} && (2.5)
\end{aligned}$$

then

$$\begin{aligned}
\Delta_{n+1}^2 &\approx \Delta_{n+1}a_{1,n}\Delta_n && (2.6)\\
&\approx R(a_{1,n})\,\Delta_{n+1}\Delta_n && (2.7)\\
&\approx R(a_{1,n})\,a_{1,n}\Delta_n^2 && (2.8)\\
&\approx \zeta_n\Delta_n^2 && (2.9)
\end{aligned}$$

We also have

$$\begin{aligned}
\Delta_{n+1}^2 &\approx R\left(\Delta_{n+1}^2\right) && (2.10)\\
&\approx R\left(\zeta_n\Delta_n^2\right) && (2.11)\\
&\approx R\left(R(a_{1,n})\,a_{1,n}\Delta_n^2\right) && (2.12)\\
&\approx R\left(\Delta_n^2\right)R(a_{1,n})\,R(R(a_{1,n})) && (2.13)\\
&\approx \Delta_n^2 R(a_{1,n})\,a_{1,n} && (2.14)\\
&\approx \Delta_n^2\zeta_n && (2.15)
\end{aligned}$$

which proves the proposition. □

Corollary 2.1.2 *Using proposition 2.1.1 inductively, it follows that for any integer k*

$$\Delta_{n+1}^{2k} \approx \zeta_n^k \Delta_n^{2k} \approx \Delta_n^{2k} \zeta_n^k \tag{2.16}$$

We may also show that the ζ_n commute with each other and their inverses in proposition 2.1.3.

Proposition 2.1.3 *If $\zeta_n = R(a_{1,n})\, a_{1,n}$, then $\zeta_i \zeta_j^{\pm 1} \approx \zeta_j^{\pm 1} \zeta_i$ for all i, j.*

Proof. We have

$$\begin{aligned}
\zeta_{n+1}\zeta_n &\approx R(a_{1,n+1})\, a_{1,n+1} R(a_{1,n})\, a_{1,n} && (2.17)\\
&\approx R(a_{1,n+1})\, \sigma_1 \cdots \sigma_{n-1}\sigma_n\sigma_{n+1}\sigma_n\sigma_{n-1} \cdots \sigma_1 a_{1,n} && (2.18)\\
&\approx R(a_{1,n+1})\, \sigma_1 \cdots \sigma_{n-1}\sigma_{n+1}\sigma_n\sigma_{n+1}\sigma_{n-1} \cdots \sigma_1 a_{1,n} && (2.19)\\
&\approx R(a_{1,n+1})\, \sigma_{n+1}\sigma_1 \cdots \sigma_{n-1}\sigma_n\sigma_{n-1} \cdots \sigma_1\sigma_{n+1} a_{1,n} && (2.20)\\
&\approx R(a_{1,n+1})\, \sigma_{n+1} a_{1,n} R(a_{1,n})\, \sigma_{n+1} a_{1,n} \text{ by induction:} && (2.21)\\
&\approx R(a_{1,n+1})\, \sigma_{n+1} \cdots \sigma_3\sigma_1\sigma_2\sigma_1\sigma_3 \cdots \sigma_{n+1} a_{1,n} && (2.22)\\
&\approx R(a_{1,n+1})\, \sigma_{n+1} \cdots \sigma_3\sigma_2\sigma_1\sigma_2\sigma_3 \cdots \sigma_{n+1} a_{1,n} && (2.23)\\
&\approx R(a_{1,n+1})\, R(a_{1,n+1})\, \sigma_1^{-1} a_{1,n+1} a_{1,n} && (2.24)\\
&\approx \sigma_{n+1} \cdots \sigma_1\sigma_{n+1} \cdots \sigma_1\sigma_1^{-1} a_{1,n+1} a_{1,n} && (2.25)\\
&\approx \sigma_{n+1}\sigma_n\sigma_{n+1}\sigma_{n-1} \cdots \sigma_1\sigma_n \cdots \sigma_1\sigma_1^{-1} a_{1,n+1} a_{1,n} && (2.26)\\
&\approx \sigma_n\sigma_{n+1}\sigma_n\sigma_{n-1} \cdots \sigma_1\sigma_n \cdots \sigma_1\sigma_1^{-1} a_{1,n+1} a_{1,n} && (2.27)\\
&\approx \sigma_n R(a_{1,n+1})\, R(a_{1,n})\, \sigma_1^{-1} a_{1,n+1} a_{1,n} \text{ by induction:} && (2.28)\\
&\approx R(a_{1,n})\, R(a_{1,n+1})\, a_{1,n+1} a_{1,n} && (2.29)\\
&\approx R(a_{1,n})\, R(a_{1,n+1})\, \sigma_1 \cdots \sigma_{n+1}\sigma_1 \cdots \sigma_n && (2.30)\\
&\approx R(a_{1,n})\, R(a_{1,n+1})\, \sigma_1\sigma_2\sigma_1\sigma_3 \cdots \sigma_{n+1}\sigma_2 \cdots \sigma_n && (2.31)\\
&\approx R(a_{1,n})\, R(a_{1,n+1})\, \sigma_2\sigma_1\sigma_2\sigma_3 \cdots \sigma_{n+1}\sigma_2 \cdots \sigma_n && (2.32)\\
&\approx R(a_{1,n})\, R(a_{1,n+1})\, \sigma_2 a_{1,n+1} \sigma_1^{-1} a_{1,n} \text{ by induction:} && (2.33)\\
&\approx R(a_{1,n})\, R(a_{1,n+1})\, \sigma_1^{-1} a_{1,n+1} a_{1,n+1} && (2.34)\\
&\approx R(a_{1,n})\, \sigma_{n+1} \cdots \sigma_3\sigma_2\sigma_1\sigma_1^{-1}\sigma_1\sigma_2\sigma_3 \cdots \sigma_{n+1} a_{1,n+1} && (2.35)\\
&\approx R(a_{1,n})\, \sigma_{n+1} \cdots \sigma_3\sigma_1\sigma_2\sigma_1\sigma_3 \cdots \sigma_{n+1} a_{1,n+1} && (2.36)\\
&\approx R(a_{1,n})\, \sigma_1\sigma_{n+1} \cdots \sigma_1\sigma_3 \cdots \sigma_{n+1} a_{1,n+1} \text{ by induction:} && (2.37)\\
&\approx R(a_{1,n})\, a_{1,n} R(a_{1,n+1})\, a_{1,n+1} && (2.38)\\
&\approx \zeta_n\zeta_{n+1} && (2.39)
\end{aligned}$$

From the definition of ζ_n, we have

$$\begin{aligned}
\zeta_{n+k}\zeta_n &\approx \sigma_{n+k}\sigma_{n+k-1} \cdots \sigma_{n+2}\zeta_{n+1}\sigma_{n+2}\sigma_{n+3} \cdots \sigma_{n+k}\zeta_n && (2.40)\\
&\approx \sigma_{n+k}\sigma_{n+k-1} \cdots \sigma_{n+2}\zeta_{n+1}\zeta_n\sigma_{n+2}\sigma_{n+3} \cdots \sigma_{n+k} && (2.41)\\
&\approx \sigma_{n+k}\sigma_{n+k-1} \cdots \sigma_{n+2}\zeta_n\zeta_{n+1}\sigma_{n+2}\sigma_{n+3} \cdots \sigma_{n+k} && (2.42)\\
&\approx \zeta_n\sigma_{n+k}\sigma_{n+k-1} \cdots \sigma_{n+2}\zeta_{n+1}\sigma_{n+2}\sigma_{n+3} \cdots \sigma_{n+k} && (2.43)\\
&\approx \zeta_n\zeta_{n+k} && (2.44)\\
& && (2.45)
\end{aligned}$$

since the highest generator contained in ζ_n is σ_n and $\sigma_i\sigma_j \approx \sigma_j\sigma_i$ for $|i-j| > 1$. Thus we have $\zeta_i\zeta_j \approx \zeta_j\zeta_i$ for all i, j. From this it follows that $\zeta_i \approx \zeta_j\zeta_i\zeta_j^{-1}$ and $\zeta_j^{-1}\zeta_i \approx \zeta_i\zeta_j^{-1}$. Hence all the ζ_i commute with themselves and their inverses and the proposition is proven. □

Proposition 2.1.4 *If we define $\zeta_0 = e$, then $\zeta_i = \sigma_i\zeta_{i-1}\sigma_i$. Furthermore, we have $\sigma_i\zeta_j \approx \zeta_j\sigma_i$ for $i < j$ or $i > j+1$.*

Proof. By definition, $\zeta_i = R(a_{1,i})\, a_{1,i}$ and thus $\zeta_i = \sigma_i \zeta_{i-1} \sigma_i$ for all $i > 2$. If we define $\zeta_0 = e$, then $\zeta_1 = \sigma_1^2 = \sigma_1 \zeta_0 \sigma_1$ and the first claim is proved.

Clearly, $\sigma_i \zeta_j \approx \zeta_j \sigma_i$ when $i > j+1$ since $\sigma_i \sigma_j \approx \sigma_j \sigma_i$ for $|i-j| > 1$. For the case $i = j-1$, we have

$$\begin{aligned}
\sigma_j \zeta_{j+1} &\approx \sigma_j \sigma_{j+1} \sigma_j \zeta_{j-1} \sigma_j \sigma_{j+1} && (2.46)\\
&\approx \sigma_{j+1} \sigma_j \sigma_{j+1} \zeta_{j-1} \sigma_j \sigma_{j+1} && (2.47)\\
&\approx \sigma_{j+1} \sigma_j \zeta_{j-1} \sigma_{j+1} \sigma_j \sigma_{j+1} && (2.48)\\
&\approx \sigma_{j+1} \sigma_j \zeta_{j-1} \sigma_j \sigma_{j+1} \sigma_j && (2.49)\\
&\approx \zeta_{j+1} \sigma_j && (2.50)
\end{aligned}$$

And for $i < j-1$,

$$\begin{aligned}
\sigma_i \zeta_j &\approx \sigma_i \sigma_j \sigma_{j-1} \cdots \sigma_{i+2} \zeta_{i+1} \sigma_{i+2} \sigma_{i+3} \cdots \sigma_j && (2.51)\\
&\approx \sigma_j \sigma_{j-1} \cdots \sigma_{i+2} \sigma_i \zeta_{i+1} \sigma_{i+2} \sigma_{i+3} \cdots \sigma_j && (2.52)\\
&\approx \sigma_j \sigma_{j-1} \cdots \sigma_{i+2} \zeta_{i+1} \sigma_i \sigma_{i+2} \sigma_{i+3} \cdots \sigma_j && (2.53)\\
&\approx \sigma_j \sigma_{j-1} \cdots \sigma_{i+2} \zeta_{i+1} \sigma_{i+2} \sigma_{i+3} \cdots \sigma_j \sigma_i && (2.54)\\
&\approx \zeta_j \sigma_i && (2.55)
\end{aligned}$$

This proves the proposition. □

2.1.2 Word Problems

As shown above, any braid can be transformed into the form $\Delta_n^{-k} P$ where P is a positive braid and k a positive integer. It is trivial to extend this to the form $\Delta_n^{-2k} P'$, where $P' = \Delta_n^k P$. Since Δ_n^2 generates the center of B_n, it may be simpler to solve the word, conjugacy and possibly Markov problems for braids in this form. The following proposition lends more weight to this intuitive judgment.

Proposition 2.1.5 *For any group G and any α, $\beta \in G$ we have $\alpha \approx_c \beta$ if and only if $\gamma\alpha \approx_c \gamma\beta$ where $\gamma \in C(G)$ where $C(G)$ is the center of G.*

Proof. If $\alpha \approx_c \beta$, then we may put $\alpha \approx \eta\beta\eta^{-1}$ for some word $\eta \in G$ by the definition of conjugacy. Due to the definition of the inverse, we have $\alpha\alpha^{-1} \approx e$, where $e \in G$ is the identity. By replacing the inverse α^{-1} with $\alpha^{-1} \approx \eta\beta^{-1}\eta^{-1}$, we have $\alpha\eta\beta^{-1}\eta^{-1} \approx e$. Concatenate $\gamma \in C(G)$ to the front of both α and β, i.e. $\alpha \to \gamma\alpha$ and $\beta \to \gamma\beta$, we obtain

$$\begin{aligned}
\gamma\alpha\eta\beta^{-1}\gamma^{-1}\eta^{-1} &\approx \gamma\gamma^{-1}\alpha\eta\beta^{-1}\eta^{-1} && (2.56)\\
&\approx \alpha\eta\beta^{-1}\eta^{-1} && (2.57)\\
&\approx e && (2.58)
\end{aligned}$$

and thus $\gamma\alpha \approx \eta\gamma\beta\eta^{-1}$ by virtue of the fact that any member of center of a group commutes with any member of the group. Thus we have proven that if $\alpha \approx_c \beta$, then $\gamma\alpha \approx_c \gamma\beta$.

To prove the converse, assume that $\gamma\alpha \approx_c \gamma\beta$ and put $\alpha' = \gamma\alpha$ and $\beta' = \gamma\beta$. Therefore, $\alpha' \approx_c \beta'$ and we may again put $\alpha' = \eta\beta'\eta^{-1}$ and therefore restate the identity $\alpha'{\alpha'}^{-1} \approx e$ as $\alpha'\eta{\beta'}^{-1}\eta^{-1} \approx e$. Thus,

$$\begin{aligned}
\alpha'\eta{\beta'}^{-1}\eta^{-1} &\approx \gamma\alpha\eta\beta^{-1}\gamma^{-1}\eta^{-1} && (2.59)\\
&\approx \alpha\eta\beta^{-1}\eta^{-1} && (2.60)\\
&\approx e && (2.61)
\end{aligned}$$

from the above. We have proven that if $\gamma\alpha \approx_c \gamma\beta$, then $\alpha \approx_c \beta$. Combining both statements, we have proven the proposition. □

If we are given two words $\alpha = \Delta_n^{-2k} P$ and $\beta = \Delta_n^{-2k'} P'$, then by proposition 2.1.5 the word or conjugacy problem between α and β may be decided by deciding it between the positive braid

words P and $\Delta_n^{2(k-k')}P'$ for $k' \leq k$. It is natural to ask whether this property extends to Markov equivalence. We ask: Is it true that for any $\alpha, \beta \in B_n$, we have $\alpha \approx_M \beta$ if and only if $\Delta_n^2\alpha \approx_M \Delta_n^2\beta$?

Note that this is enough. Repeated application of the statement would show that $\Delta_n^{2k}\alpha \approx_M \Delta_n^{2k}\beta$ if and only if $\alpha \approx_M \beta$ for any k, possibly negative. Note that it is not true, for any braids $\alpha, \beta, \gamma \in B_n$, that $\alpha \approx_M \beta$ if and only if $\gamma\alpha \approx_M \gamma\beta$ for $\gamma \in C(B_n)$. To see this, recall that $\approx_c$ is a subequivalence of $\approx_M$ and $\alpha \approx_c \beta$ is only true if and only if $\gamma\alpha \approx_c \gamma\beta$ for any $\gamma \in C(B_n)$.

2.1.3 The Peripheral Group System

In this section we will investigate the peripheral group system of the closure of the fundamental braids. This is interesting in its own right and will illustrate the only known complete invariant of knots. The closure of Δ_3 is the Hopf Link and the closures of the other fundamental braids look very similar to Hopf Links. In fact so similar that we can regard the class of knots defined by the closure of fundamental braids as a generalization of the Hopf Link. Another generalization of the Hopf link has been investigated in the literature [40].

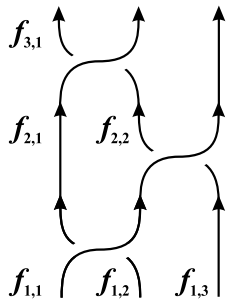


Figure 2.1: The braid Δ_3 with labels for the Wirtinger representation of the complement of its closure.

Consider the braid $\Delta_3 = \sigma_1\sigma_2\sigma_1$, see figure 2.1. The closure of this braid is the Hopf Link and we wish to find its peripheral group system. There are six line segments in the diagram and we label them as in the figure. Closure identifies a number of the segments to give us,

$$f_{1,3} = f_{3,1} = f_{1,1}, \qquad f_{1,2} = f_{2,1} \tag{2.62}$$

and the three crossings, by Wirtinger's method described in chapter 1, give rise to the relations

$$f_{1,2} = f_{1,1}^{-1}f_{1,2}f_{1,1}, \qquad f_{1,3} = f_{1,1}^{-1}f_{2,1}^{-1}f_{3,1}f_{2,1}f_{1,1}, \qquad f_{2,2} = f_{2,1}^{-1}f_{3,1}f_{2,1} \tag{2.63}$$

After some manipulation, we obtain

$$f_{1,3} = f_{3,1} = f_{2,2} = f_{1,1} \tag{2.64}$$
$$f_{1,2} = f_{2,1} \tag{2.65}$$
$$f_{1,1}f_{2,1} = f_{2,1}f_{1,1} \tag{2.66}$$

So if we put $a = f_{1,1}$ and $b = f_{2,1}$ we get

$$\pi_1\left(\overline{\Delta_3}\right) = \langle \{a, b\} : [a, b] \rangle \tag{2.67}$$

We may select any generator from the relevant component as its meridian and any path through the component as the longitude, so the meridian-longitude system pair $\mathcal{M}\left(\overline{\Delta_3}\right)$ is $(\{a, b\}, \{a, b\})$ (recall the definition from the discussion of figure 1.4). Due to the fact that the Hopf Link has two components, the fundamental group must have at least two generators and so this is a minimal presentation. As mentioned above, $\pi_1\left(\overline{\Delta_3}\right)$ together with $\mathcal{M}\left(\overline{\Delta_3}\right)$ characterizes the Hopf Link up to isotopy.

We note in particular that we needed only a single generator per component in the above example. In an effort to see how robust this property is, we compute the peripheral group system

for a few Δ_n. When the set of meridians and the set of longitudes are both equal to the set of generators of the fundamental group of the complement of a knot K, we speak of a *trivial meridian-longitude system pair* and denote it by $\mathcal{T}(K)$, we omit the argument when no confusion can arise. The labeling in figure 2.1 trivially extends to arbitrary n and we obtain,

$$\begin{aligned}
p\left(\overline{\Delta_1}\right) &= \left(\langle\{a\}\rangle;\mathcal{T}\right) && (2.68)\\
p\left(\overline{\Delta_2}\right) &= \left(\langle\{a\}\rangle;\mathcal{T}\right) && (2.69)\\
p\left(\overline{\Delta_3}\right) &= \left(\langle\{a,b\}:[a,b]\rangle;\mathcal{T}\right) && (2.70)\\
p\left(\overline{\Delta_4}\right) &= \left(\left\langle\{a,b\}:(ab)^2=(ba)^2\right\rangle;\mathcal{T}\right) && (2.71)\\
p\left(\overline{\Delta_5}\right) &= \left(\langle\{a,b,c\}:[a,b],[a,c],[b,c]\rangle;\mathcal{T}\right) && (2.72)\\
p\left(\overline{\Delta_6}\right) &= \left(\left\langle\{a,b,c\}:(acb)^2=(bac)^2=(cba)^2\right\rangle;\mathcal{T}\right) && (2.73)\\
p\left(\overline{\Delta_3^2}\right) &= \left(\langle\{a,b,c\}:[a,b],[a,c],[b,c]\rangle;\mathcal{T}\right) && (2.74)
\end{aligned}$$

where we have assigned the $f_{i,1}$ to consecutive letters of the alphabet. Also, due to the definition of Δ_n, we have $\Delta_1 = \underline{e}$ and $\Delta_2 = \sigma_1$. Aside from obvious patterns, we have thus shown that $\overline{\Delta_5}$ is ambient isotopic to $\overline{\Delta_3^2}$ (see figure 2.2).

Proposition 2.1.6 *Let $S = \{g_i\}$ for $1 \leq i \leq m$, $G = g_1 g_2 \cdots g_m$ and G^c be the c^{th} cyclic permutation of G,*

$$G^c = g_{c+1} g_{c+2} \cdots g_m g_1 g_2 \cdots g_c \tag{2.75}$$

We have $G \approx G^c$ for all c if and only if $[g_i, g_j]$ for all i and j.

Proof. If $g_i g_j = g_j g_i$ for all i and j, then $G \approx G^c$ is obvious, so we need to prove the converse. For $m = 1, 2$ the result is trivially true. Consider

$$g_c G^c g_{c+1} = G^{c-1} g_c g_{c+1} = g_c g_{c+1} G^{c+1} \tag{2.76}$$

but $G^i \approx G^j$ for all i and j by assumption. Hence $[g_i, G^j]$ and thus $[g_i, g_j]$ for all i and j. □

The free Abelian group $\mathcal{C}_\infty^k$ of rank k is given by the direct product of k copies of the infinite cyclic group on one element. Let $\mathcal{C}_\infty$ be the infinite cyclic group on one element, then $\mathcal{C}_\infty =< \{a\} >$ [51]. The k^{th} direct product

$$\mathcal{C}_\infty^k = \underbrace{\mathcal{C}_\infty \times \mathcal{C}_\infty \times \cdots \times \mathcal{C}_\infty}_{k \text{ times}} \tag{2.77}$$

has presentation

$$\mathcal{C}_\infty^k = \langle \{g_i\} \text{ for } 1 \leq i \leq k : \ [g_i, g_j] \ \forall i, j \rangle \tag{2.78}$$

Comparing this with the information in equations 2.68 to 2.74 suggests that the peripheral group system of the closure of fundamental braids is the free Abelian group of rank n with trivial meridian-longitude system for odd n.

In figure 2.2, we demonstrate how the labeling is to be extended for a product of several Δ_3. This generalizes in an obvious way to Δ_n^p, which we consider now. Through double-labeling some segments, we have

$$f_{jn+1,k} = f_{jn+1-k,1} \tag{2.79}$$

for $1 \leq j < p$ and $1 \leq k \leq n$ and closure gives

$$f_{1,k} = f_{pn+1-k,1} \tag{2.80}$$

The relations due to crossings are

$$f_{r,c} = f_{r,1}^{-1} f_{r+1,c-1} f_{r,1} \tag{2.81}$$

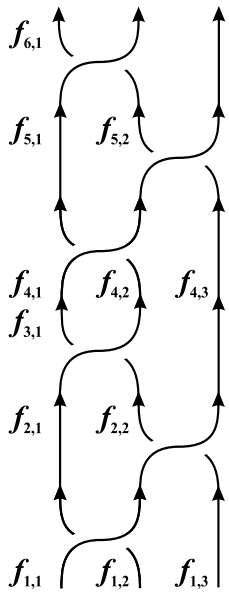


Figure 2.2: The braid Δ_3^2 with labels for the Wirtinger representation of the complement of its closure.

by using (2.81) recursively, we find that

$$f_{r,c} = \prod_{k=0}^{c-2} f_{r+k,1}^{-1} \prod_{k=c-1}^{0} f_{r+k,1} \tag{2.82}$$

for $1 \leq r \leq pn$ and $1 \leq c \leq n - [r-1 \mod n]$. We may use relations (2.82) to eliminate all $f_{r,c}$ with $c > 1$ in terms of $f_{r,1}$. For convenience we put $g_r = f_{r,1}$. Equations (2.79) and (2.80) may be written in one and combined with (2.82) to give a compact description of the fundamental group of the complement of Δ_n^p

$$\pi_1\left(\overline{\Delta_n^p}\right) = \left\langle \{g_i\} 1 \leq i \leq pn : \; g_{jn-c} = \prod_{k=1}^{c} g_{r+k}^{-1} \prod_{k=c+1}^{1} g_{r+k} \right\rangle \tag{2.83}$$

where $r = (j \mod p)n$, $1 \leq j \leq p$ and $0 \leq c \leq n-1$.

2.2 A Review of Term Rewriting Systems

The following presentation of the theory of term rewriting systems is intentionally simplified and incomplete; only results which are of direct relevance to the new material are presented. A more complete treatment of the theory may be found in the monograph [9] and the review papers [56], [95], [84].

We begin with a finite alphabet of *constants* $\mathcal{A}$ and a finite set of *variables* $\mathcal{X}$. A *term* t is a finite ordered sequence of constants and variables $t = a_1 a_2 \cdots a_n$ with $n \geq 0$ (i.e. empty terms are allowed) and $a_i \in \mathcal{A} \cup \mathcal{X}$. A *word* w is a finite ordered sequence of constants $w = b_1 b_2 \cdots b_m$ with $m \geq 0$ and $b_i \in \mathcal{A}$. A *substitution* ρ for a term t is a map which assigns a word to each variable in t; the resultant word is denoted by ρt. A *term rewriting system* (TRS) $\mathcal{R} = \{(l_i, r_i)\}$ is a set of ordered pairs of terms l_i and r_i. Each ordered pair in $\mathcal{R}$ is referred to as a *rule* or *rewrite rule* and is often written in the form $l_i \to r_i$; the whole TRS is sometimes denoted by $\to_{\mathcal{R}}$. A TRS $\mathcal{R} = \{(l_i, r_i)\}$ is applied to a word w_0 by determining if w_0 contains the word ρl_i, for some substitution ρ, as a subword. If and only if w_0 contains ρl_i is ρl_i replaced by ρr_i. If $\to$ is a rewrite rule, then $\leftarrow$ is its inverse, $\leftrightarrow$ is its symmetric closure ($\leftarrow \cup \to$) and $\to^\star$ is its reflexive-transitive closure ($\to \circ \to \circ \cdots \circ \to$). Two terms t and s are said to be *joinable* if there exists a term r such

that $t \rightarrow^\star r \leftarrow^\star s$. Any l_i is called a *redex* and any r_i is called a *reduct* (these are abbreviations of reducible expression and reduced term).

A word w_0 is thus rewritten into a word w_1 if and only if $\mathcal{R}$ may be applied to w_0. We may generate a *rewrite chain* of words $w_0 \rightarrow_\mathcal{R} w_1 \rightarrow_\mathcal{R} \cdots$ in this manner. $\mathcal{R}$ *terminates* if and only if there exists no rewrite chain of infinite length. $\mathcal{R}$ is *locally confluent* if and only if any local divergence $\leftarrow \circ \rightarrow$ is contained in the joinability relation $\rightarrow^\star \circ \leftarrow^\star$. $\mathcal{R}$ is *confluent* if and only if any divergence $\leftarrow^\star \circ \rightarrow^\star$ is contained in the joinability relation $\rightarrow^\star \circ \leftarrow^\star$. $\mathcal{R}$ is *complete* if it is confluent and terminates. If $\mathcal{R}$ is complete a unique normal form exists for each word [9]; the final form obtained by applying $\mathcal{R}$ to the word a maximum number of times.

It should be noted that the computational power of term rewriting systems is identical to that of Turing machines, i.e. one may be simulated by the other [149]. According to the Church-Turing thesis [47], this means that any function which may reasonably be termed computable is computable using a TRS.

2.2.1 Word Problems

Consider a group G with presentation $G = \langle X, E\rangle$ for some finite set of generators $X = \{f_i\}$ and a finite set of equations E. A word in the group is then a sequence of the generators and their inverses f_i^{-1}. It is possible to construct a large (possibly infinite) number of words in G which are all equivalent to the empty word e under the set of equations E. It was first proposed by Dehn [53] that the question, known as the *word problem*, whether $w \approx_E e$ for any word $w \in G$ is interesting. Note that the question of equality, $w_1 \approx_E w_2$ for any two words $w_1, w_2 \in G$ is contained in the word problem as $w_1 \approx_E w_2$ if and only if $w_1 w_2^{-1} \approx_E e$. Two related problems, also suggested by Dehn, are the *conjugacy problem* in which one decides if there exists a $w_3 \in G$ such that $w_1 \approx_E w_3 w_2 w_3^{-1}$ and the *isomorphism problem* in which one decides if a presentation of some group $G' = \langle \{f_i'\}, E'\rangle$ represents the same group as G. All three are, in general, unsolvable.

It was proven by Birkhoff [19] that the symmetric-reflexive-transitive closure $\leftrightarrow^\star_\mathcal{R}$ of a TRS $\mathcal{R} = \{(l_i, r_i)\}$ is equivalent to the set of equations $\mathcal{E} = \{l_i = r_i\}$. It is an obvious corollary to Birkhoff's theorem that if there exists a complete TRS $\mathcal{R}$ over the alphabet $\mathcal{A} = \{f_i, f_i^{-1}\}$ for which $\leftrightarrow^\star_\mathcal{R}$ contains exactly the equations $\{\mathcal{E}, f_i f_i^{-1} = e, f_i^{-1} f_i = e\}$, then $\mathcal{R}$ solves the word problem for the group $G = \langle \{f_i\}, E\rangle$.

Note that $\mathcal{R}$ also solves the word problem for the monoid associated with G, i.e. the monoid obtained when the inverses of the generators are added to the set of generators and the fact that the generators and inverses are in fact inverses ($f_i f_i^{-1} = e, f_i^{-1} f_i = e$) added to the set of equations. It can be shown that the solubility of the word problem does not depend on the presentation in the case of a group [110] but does depend on the presentation in the case of a monoid (recall that a general monoid need not have inverses) [31] [12]. However there exist groups, and hence monoids, for which the word problem is not solvable [121] [32]. This fact is a special case of the Higman Embedding Theorem [79] which asserts that if a finitely generated group has a recursively enumerable set of relations, then it can be embedded in a finitely presented group. It can be shown that this embedding preserves the solubility of the word problem [45] but not the conjugacy problem [49]. This gives an immediate source of groups with solvable word problem and unsolvable conjugacy problem.

All mathematical unsolvability results can be traced to the fact that there exist recursively enumerable sets (there exists an algorithm to list all elements) which are not recursive (there exists an algorithm for testing membership) [110]. Since the conjugacy problem includes the word problem (put $w_3 = e$ in the discussion above) a similar result exists for it with the notable exception of all groups with a single relation for which there exists an algorithm for the conjugacy and hence word problems [88]. Moreover, the isomorphism problem is, in general, unsolvable also [2] [3] [129]. The situation is, in fact, even worse as there exists no algorithm which solves the word problem in all groups with solvable word problem (this holds for most of the interesting group properties [13]) and the problem of determining whether a group has solvable word problem is Σ_3^0-complete [33] [103] in the Kleene (or Grzegorczyk) arithmetical hierarchy [63] (that is the problem is much harder than an NP-complete problem, if P $\neq$ NP). For a good survey on what is and what is not possible see [110] and [109], for background on other methods of combinatorial group theory see [102] and [51].

2.2.2 Termination

It is, in general, undecidable whether a TRS terminates or not [82]. Since any Turing machine can be modeled using a TRS this is essentially due to the undecidability of whether a Turing machine will stop, the Turing Halting Problem [150]. It is however decidable for a TRS without any variables [55]. Thus, in general, a termination proof is specific to a particular TRS and must be given for it. A common strategy for proving termination is to use a reduction order on the symbols involved in the TRS. We define a *reduction order* $<_o$ as a strict order over the alphabet and variables of the TRS which satisfies:

1. **compatibility**: For all terms u, v for which $u <_o v$, we have $xuy <_o xvy$ for any terms x and y.
2. **closure**: For all terms u, v for which $u <_o v$ and all substitutions σ, we have $\sigma u <_o \sigma v$.
3. **basis**: $<_o$ is well-founded, i.e. there exists a simplest term under $<_o$.

If one can show that every possible rewriting operation simplifies any term with respect to such an ordering, then the TRS terminates [54]. Furthermore, a TRS $\mathcal{R}$ terminates if and only if there exists a reduction order $<_o$ which satisfies $r_i <_o l_i$ for every rule $l_i \to r_i \in \mathcal{R}$ [9]. This is true because every step of the rewriting process simplifies the term and there exists a simplest term. Another useful result is that a TRS terminates if and only if it terminates for all instances of its redexes [57]. Some conditions under which the union of two terminating TRSs is terminating are analyzed in [57].

2.2.3 Confluence

Like termination, confluence is, in general, undecidable [9]. However, for terminating systems there exists a mechanizable method for deciding confluence [80] that rests on Newman's Lemma which states that a terminating TRS is confluent if and only if it is locally confluent [120] (we shall prove a generalization of this in lemma 2.3.5). Local confluence can be decided by a systematic method which searches for critical pairs in the TRS. The concept of critical pairs is difficult to trace in history; for an attempt at a historical survey see [38] and for a good technical treatment see [80].

Given a TRS $\mathcal{R} = \{(l_i, r_i)\}$, an *overlap* is a word $w = abc$ such that $ab = \rho l_i$ and $bc = \eta l_j$ for some words a, b and c, two (possibly equal) integers i and j and substitutions ρ and η. Clearly the overlap abc may be rewritten to both $\rho r_i c$ and $a\eta r_j$. An overlap is *non-critical* if the reducts are joinable, $\rho r_i c \leftrightarrow^\star_{\mathcal{R}} a\eta r_j$ and *critical* otherwise. A *critical pair* is the (unordered) pair $(\rho r_i c, a\eta r_j)$ which arises from a critical overlap. It is obvious that if $\mathcal{R}$ contains critical pairs, it can not be confluent. The fact that the non-existence of critical pairs is both a necessary and sufficient condition for local confluence is called the Critical Pair Lemma [87]. Later we shall prove lemma 2.3.6 which contains the Critical Pair Lemma.

2.2.4 Completeness

If we can find a reduction order for a TRS $\mathcal{R}$, thereby prove termination and find that there are no critical pairs, $\mathcal{R}$ is complete and thus solves the word problem for $\leftrightarrow^\star_{\mathcal{R}}$. A general procedure for what to do when we can not do this is called Knuth-Bendix completion from their seminal paper [96]. Again a historical account of this procedure is tangled and [38] is an attempt to unravel it. We shall follow the common practice to call it Knuth-Bendix completion even though, by their own admittion, the initial idea was not theirs.

Suppose we have a set of equations $\mathcal{E}$ on an alphabet $\mathcal{A}$ and a total order $<_{\mathcal{A}}$ (this is a reduction order) on $\mathcal{A}$. Construct a TRS $\mathcal{R}$ from $\mathcal{E}$ by creating a rule $l_i \to r_i$ in $\mathcal{R}$ from the equation $l_i = r_i$ in $\mathcal{E}$ for all equations in $\mathcal{E}$ such that the rules are ordered such that $l_i >_{\mathcal{A}} r_i$. Now $\leftrightarrow^\star_{\mathcal{R}}$ is equivalent to $\mathcal{E}$ and each rule represents a simplification in terms of $<_{\mathcal{A}}$. Clearly there exists a simplest word, the empty word, and so $\mathcal{R}$ terminates.

If there are no critical pairs, $\mathcal{R}$ is locally confluent and thus complete. If there are critical pairs, order them with respect to $<_{\mathcal{A}}$ and append them to $\mathcal{R}$ as new rules. Termination still holds and so we continue this process. We may delete duplicate or redundant rules from $\mathcal{R}$ between the steps of this method to obtain a smaller TRS. If this method terminates, we have a complete

system [96] [81]. If it does not terminate, a complete system may still exist which contains an infinite number of rules. It is possible to collect an infinite number of rules into a finite number of rules by introducing variables. The Knuth-Bendix algorithm has been implemented by several people and can be used to determine, in some cases, whether a complete system exists. The CiME implementation was used for this thesis [104]. Producing rules with variables and proving the non-existence of critical pairs is, at present, beyond the computer implementations and must be done manually.

The process described here is simplified; there are more pitfalls, in general, and the method has been considerably extended to take into account many other features (many relevant references are in [38]). The method as described is enough for our purposes here however and is generally enough for a word problem in a finitely presented group.

2.3 Word and Conjugacy in B_n

Having reviewed TRS's, we are now in a position to find a TRS for the word problem in B_n. The braid group B_n is defined formally as

$$\begin{aligned} B_n = \langle \{\sigma_1, \sigma_2, \cdots, \sigma_{n-1}\} : \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}; \\ \sigma_i\sigma_j = \sigma_j\sigma_i \text{ for } |i-j| > 1 \rangle \end{aligned} \tag{2.84}$$

Given a finitely presented group $G = \langle X, E\rangle$, we can define an associated monoid $M(G) = \left\langle X \cup X^{-1}, E \cup aa^{-1} = 1\right\rangle$ for any $a \in X$. It is clear that the equivalence and conjugacy classes of the group G and the monoid $M(G)$ are identical. In order to solve the word problem for B_n, we augment the monoid $M(B_n)$ with the generator of the center of B_n, Δ_n^2 to form the monoid

$$\begin{aligned} M^+(B_n) = \langle \{\sigma_1^{\pm 1}, \sigma_2^{\pm 1}, \cdots, \sigma_{n-1}^{\pm 1}, \Delta_n^{\pm 2}\} : \Delta_n^{\pm 2}\sigma_i = \sigma_i\Delta_n^{\pm 2}; \\ \Delta_n^{\pm 2}\Delta_n^{\mp 2} = \sigma_i^{\pm 1}\sigma_i^{\mp 1} = e; \\ \sigma_i^{\pm 1}\sigma_j^{\pm 1/\mp 1} = \sigma_j^{\pm 1/\mp 1}\sigma_i^{\pm 1} \text{ for } |i-j| > 1; \\ \sigma_i^{\pm 1}\sigma_{i+1}^{\pm 1}\sigma_i^{\pm 1} = \sigma_{i+1}^{\pm 1}\sigma_i^{\pm 1}\sigma_{i+1}^{\pm 1} \rangle \end{aligned} \tag{2.85}$$

It is obvious from the definition of the monoid $M^+(B_n)$ that a solution of its word and conjugacy problems provides a solution for the word and conjugacy problems in the group B_n.

2.3.1 The Word Problem in B_n

We will use Knuth-Bendix completion upon the oriented rules of $M^+(B_n)$ under the reduction order $<_b$

$$\Delta_n^2 <_b \Delta_n^{-2} <_b \sigma_1 <_b \sigma_2 <_b \cdots <_b \sigma_{n-1} <_b \sigma_1^{-1} <_b \sigma_2^{-1} <_b \cdots <_b \sigma_{n-1}^{-1} \tag{2.86}$$

In practice, this process is laborious and would occupy prodigious space if described in detail. For this reason, we will simply state the result and prove it to be correct.

For what follows, we shall represent a braid of the form $\Delta_n^{2k}P$ as the pair (k, P). The reason for this is to effectively remove from the braid, in the process of rewriting, any subbraid which lies in the center of the braid group B_n. The reason for this will become apparent when we extend our solution to the conjugacy problem. Removing any Δ_n^{2k} from any part of a braid can be done without loss of information because Δ_n^{2k} is the generator of the center of B_n and thus its position is irrelevant. By Knuth-Bendix completion and the necessary manual labor, we obtain the following

rewriting system.

$$\begin{aligned}\mathcal{W}_n = \{\, &(1)\ \sigma_i^{-1} \to \prod_{j=1}^{i-1} [d_{j,1}a_{1,j}]\, d_{i,1}a_{1,i-1} \prod_{j=i+1}^{n-1} [d_{j,1}a_{1,j}] \ \ \&\ \ k \to k-1;\\ &(2)\ \sigma_i\sigma_j \to \sigma_j\sigma_i \text{ for } j < i-1;\\ &(3)\ \sigma_i\sigma_{i-1}P\sigma_i \to \sigma_{i-1}\sigma_i\sigma_{i-1}P;\\ &(4)\ \sigma_i\sigma_{i-1}Q\sigma_{i-1}Rd_{i,j} \to \sigma_{i-1}\sigma_i\sigma_{i-1}Qd_{i-1,j}\sigma_i R^+ \text{ for } j < i;\\ &(5)\ \prod_{i=1}^{n-1} d_{i,1}a_{1,i}S_i \to \prod_{i=1}^{n-1} S_i \ \ \&\ \ k \to k+1\ \}\end{aligned} \tag{2.87}$$

The variables P, Q, R and S_i are (possibly empty) words in the generators σ_k (and *not* their inverses σ_k^{-1}) subject to the restriction that the highest generator index k is $i-2$, $i-2$, $i-1$ and i respectively and the lowest generator index in R is j, where i and j refer to the values of the generator indices of the respective rules. The word R^+ is obtained from R by increasing all generator indices in R by one. Note that rules 1 and 5 require two replacements to be made simultaneously. A similar, unpublished TRS was also found using Knuth-Bendix completion by Yoder [161]. Rules 1 and 5 are simple to understand; the other rules are illustrated in figure 2.3.

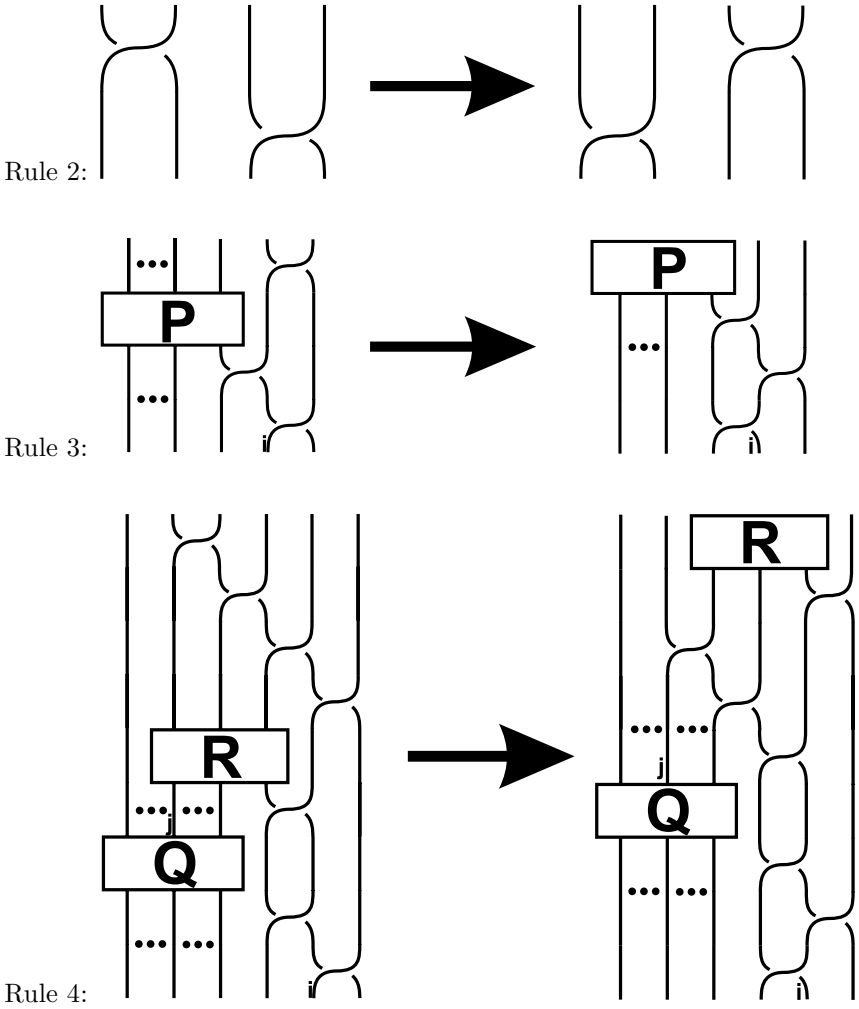


Figure 2.3: Rules 2, 3 and 4 of TRS $\mathcal{W}_n$ illustrated.

Theorem 2.3.1 *$\mathcal{W}_n$ is complete and solves the word problem for B_n.*

Table 2.1: Overlaps between rules in $\mathcal{W}_n$. (The ellipses, $\cdots$, indicate line breaks and not pattern continuation signs.)

$\rightarrow$	Overlap	Final Form
2, 2	$\sigma_i\sigma_j\sigma_k$	$\sigma_k\sigma_j\sigma_i$
2, 3	$\sigma_i\sigma_j\sigma_{j-1}P\sigma_j$	$\sigma_{j-1}\sigma_j\sigma_{j-1}P\sigma_i$
2, 3	$\sigma_i\sigma_{i-1}\sigma_jP\sigma_i$	$\sigma_j\sigma_{i-1}\sigma_i\sigma_{i-1}P$
2, 3	$\sigma_i\sigma_{i-1}P\sigma_i\sigma_j$	$\sigma_{i-1}\sigma_i\sigma_{i-1}P\sigma_j$
2, 4	$\sigma_i\sigma_j\sigma_{j-1}Q\sigma_{j-1}Rd_{j,k}$	$\sigma_{j-1}\sigma_j\sigma_{j-1}Qd_{j-1,k}\sigma_jR^+\sigma_i$
2, 4	$\sigma_i\sigma_{i-1}\sigma_kQ\sigma_{i-1}Rd_{i,j}$	$\sigma_k\sigma_{i-1}\sigma_i\sigma_{i-1}Qd_{i-1,j}\sigma_iR^+$
2, 4	$\sigma_i\sigma_{i-1}Q\sigma_{i-1}Rd_{i,j}\sigma_k$	$\sigma_{i-1}\sigma_i\sigma_{i-1}Q\sigma_kd_{i-1,j}\sigma_iR^+$
2, 5	$\prod_{i=1}^{n-1} d_{i,1}a_{1,i}S_i; S_j = \sigma_kS'_j$	$\prod_{i=1}^{n-1} S_i; S_j = \sigma_jS'_j$
3, 3	$\sigma_i\sigma_{i-1}P\sigma_i\sigma_{i-1}P'\sigma_i$	$\sigma_{i-1}\sigma_i\sigma_{i-1}P\sigma_{i-1}P'\sigma_i$
3, 4	$\sigma_i\sigma_{i-1}P\sigma_i\sigma_{i-1}Q\sigma_{i-1}Rd_{i,j}$	$\sigma_{i-1}\sigma_i\sigma_{i-1}P\sigma_{i-1}Q\sigma_{i-1}Rd_{i,j}$
3, 4	$\sigma_i\sigma_{i-1}\sigma_{i-2}P\sigma_{i-1}R\cdots$ $\cdots\sigma_{i-2}R'\sigma_{i-1}R''d_{i,j}$	$\sigma_{i-2}\sigma_{i-1}\sigma_{i-2}\sigma_i\sigma_{i-1}\sigma_{i-2}P\cdots$ $\cdots Rd_{i-2,j}\sigma_{i-1}R'^+\sigma_iR''^+$
3, 4	$\sigma_i\sigma_{i-1}\sigma_{i-2}P\sigma_{i-1}R\sigma_{i-1}R'd_{i,j}$	$\sigma_{i-2}\sigma_{i-1}\sigma_{i-2}\sigma_i\sigma_{i-1}\sigma_{i-2}PRd_{i-2,j}\sigma_iR'^+$
3, 4	$\sigma_i\sigma_{i-1}\sigma_{i-2}P\sigma_{i-1}R\sigma_{i-2}R'd_{i,j}$	$\sigma_{i-2}\sigma_{i-1}\sigma_{i-2}\sigma_i\sigma_{i-1}\sigma_{i-2}PRd_{i-2,j}\sigma_{i-1}R'^+$
3, 4	$\sigma_i\sigma_{i-1}\sigma_{i-2}P\sigma_{i-1}Rd_{i,j}$	$\sigma_{i-2}\sigma_{i-1}\sigma_{i-2}\sigma_i\sigma_{i-1}\sigma_{i-2}PRd_{i-2,j}$
3, 4	$\sigma_i\sigma_{i-1}P\sigma_{i-1}Rd_{i,j}R'\sigma_k$	$\sigma_{i-1}\sigma_i\sigma_{i-1}Pd_{i-1,j}\sigma_iR^+R'\sigma_k$
3, 5	$\prod_{i=1}^{n-1} d_{i,1}a_{1,i}S_i; S_j = \sigma_{j-1}P\sigma_jS'_j$	$\prod_{i=1}^{n-1} S_i; S_j = \sigma_{j-1}P\sigma_jS'_j$
4, 4	$\sigma_i\sigma_{i-1}Q\sigma_{i-1}Rd_{i,j}Q'\sigma_{k-1}R'd_{k,m}$	$\sigma_{i-1}\sigma_i\sigma_{i-1}Qd_{i-1,j}\sigma_iR^+Q'\sigma_{k-1}R'd_{k,m}$
4, 4	$\sigma_i\sigma_{i-1}\sigma_{i-2}Q\sigma_{i-2}Rd_{i-1,j}Q'd_{i,k}$	$\sigma_{i-2}\sigma_{i-1}\sigma_{i-2}\sigma_i\sigma_{i-1}\sigma_{i-2}Q\cdots$ $\cdots d_{i-2,k}d_{i-1,j+1}\sigma_iR^{++}Q'^+$
4, 4	$\sigma_i\sigma_{i-1}Q\sigma_{i-1}Rd_{i,j}$	$\sigma_{i-1}\sigma_i\sigma_{i-1}Qd_{i,j}\sigma_iR^+$
4, 5	$\prod_{i=1}^{n-1} d_{i,1}a_{1,i}S_i$ $S_j = \sigma_{j-1}Q\sigma_{j-1}Rd_{j,k}S'_j$	$\prod_{i=1}^{n-1} S_i$ $S_j = \sigma_{j-1}Q\sigma_{j-1}Rd_{j,k}S'_j$

Proof. *(termination)* Every application of $\mathcal{W}_n$ simplifies the word with respect to $<_b$. As $<_b$ is well-founded, $\mathcal{W}_n$ terminates.

(local confluence) There are 20 overlaps between the rules in $\mathcal{W}_n$ and none give rise to a critical pair. For reasons of space, we do not provide all the reduction steps for each overlap but list all overlaps, the rules from which they arise and the common reduct of all reduction paths of the overlap in table 2.1. The restrictions on the indices and the variables are obvious from the context and the definition of $\mathcal{W}_n$. The dedicated reader may easily but laboriously verify that the list is both complete and correct. There are an additional 16 (four variables and four positive redexes) variable overlaps, i.e. overlaps in which a variable completely contains a redex, but these resolve trivially and so are not listed in the table.

(equivalence) Rules 2 and 3 imply both braid group relations. Rule 5 represents the definition of Δ_n^2 in terms of the σ_i. The second parts of rules 1 and 5 imply that Δ_n^2 and Δ_n^{-2} are inverses. Rule 1, after use of the braid group relations, the definition of Δ_n^2 and Δ_n^{-2} indicates that σ_i and σ_i^{-1} are inverses. All (and only) the relations in the monoid $M^+(B_n)$ are thus contained in $\mathcal{W}_n$. □

The rules of a TRS are to be applied in a non-deterministic way and a complete TRS always reaches the unique normal form no matter what strategy of rule application is chosen [9]. Since $\mathcal{W}_n$ is complete and all strategies are equivalent, we will choose the following strategy.

Algorithm 2.3.2 *Input: A word $w \in B_n$. Output: A word $w' \in B_n$ which is the unique representative of the equivalence class of w.*

1. Apply rule 1 of $\mathcal{W}_n$ as many times as possible.

2. Apply rules 2, 3, 4 and 5 of $\mathcal{W}_n$ as many times as possible in order proceeding to the next rule only if the current can no longer be applied.

3. Loop step 2 until no rule may be applied to the word at all. In this case w' has been found.

It is clear that algorithm 2.3.2 solves the word problem from the completeness of $\mathcal{W}_n$ and the fact that once rule 1 is applied as many times as possible, it can not be applied again no matter what other rewrite steps follow as there will be no more inverse generators. From this algorithm, we are able to deduce the computational complexity of this word problem solution.

Theorem 2.3.3 *$\mathcal{W}_n$ solves the word problem for any word $w \in B_n$ of word length l with complexity $O\left(l^2 n^4\right)$.*

Proof. Suppose that w contains exactly m inverse generators. Rule 1 may be applied exactly m times, note that m goes as $O(l)$. We must search the word for the redexes of rule 1 and then replace them. Searching is an $O(l)$ operation but the reducts increase in length as n^2 and thus the application of rule 1 takes time $O\left(ln^2\right)$. It is clear that rule 1 may never be applied again and the word length of w is now $L(w) = l + mn(n-1) - m = O\left(ln^2\right)$ as m is of order l. Rule 2 may be applied a number of times bounded by $L(w)^2$ as it is a pairwise comparison between all generators in the word, at worst. An application of rules 3 and 4 may give rise to a further application of itself or the other rule but strictly later in the word and thus the number of times they may be applied is bounded by $L(w)$. While rules 2, 3 and 4 keep the word length constant, rule 5 reduces it by $n(n+1)$ and thus rule 5 may be applied a maximum of

$$\frac{l + mn(n-1) - m}{n(n+1)} = \frac{O\left(ln^2\right)}{O\left(n^2\right)} = O\left(l\right) \tag{2.88}$$

times. Thus the total worst-case complexity of the algorithm is $O\left(L(w)^2\right) = O\left(l^2 n^4\right)$. The application of rule 2 is responsible for the quadratic behavior; it is the bottleneck of the calculation. □

2.3.2 The General Conjugacy Problem

Recall that two words $a, b \in G$ for any group G are called *conjugate* (denoted $a \approx_c b$) if and only if there exists a $d \in G$ for which $a \approx dbd^{-1}$ where $\approx$ denotes equivalence in G. Note that there exist finitely-presented groups with solvable word problem but unsolvable conjugacy problem [48] and that the word problem is subsumed by the conjugacy problem by requiring $d = e$, the empty word.

Conjugacy in Free Groups

Suppose that $G = \mathcal{F}_n$ the free group of rank n. This group is generated by n elements $\{f_i\}$ for $1 \leq i \leq n$ and no relations [86]. A general word $w \in \mathcal{F}_n$ takes the form

$$w = f_{s_1}^{p_1} f_{s_2}^{p_2} \cdots f_{s_m}^{p_m}, \qquad 1 \leq s_k \leq n \tag{2.89}$$

Since there are no relations in $\mathcal{F}_n$, the word w is unique over its equivalence class if and only if $s_i \neq s_{i+1}$ for all i. This condition is trivially obtained from any word $w \in \mathcal{F}_n$ by applying the (obviously) complete rewriting system

$$\mathcal{R}_w\left(\mathcal{F}_n\right) = \{f_s^p f_s^q \rightarrow f_s^{p+q},\ \forall 1 \leq s \leq n\} \tag{2.90}$$

Thus $\mathcal{R}_w\left(\mathcal{F}_n\right)$ solves the word problem in any free group $\mathcal{F}_n$. Moreover, it does so in a time proportional to the length of the word w.

Consider now the conjugacy problem in $\mathcal{F}_n$. We define the i^{th} *cyclic permutation* $C^i(w)$ of a word w in the general form of equation (2.89) by

$$C^i(w) = f_{s_j}^{p_j'} \cdots f_{s_{m-1}}^{p_{m-1}} f_{s_m}^{p_m} f_{s_1}^{p_1} f_{s_2}^{p_2} \cdots f_{s_j}^{p_j''} \tag{2.91}$$

such that

$$p'_j + \sum_{k=j+1}^{m} p_k = i \tag{2.92}$$

Intuitively, the i^{th} cyclic permutation is obtained by taking the last i generators in the word w and moving them to the front of the word w without changing their relative order. We shall say that two words w and w' are *cyclicly permutable* (denoted $\approx_{cp}$) if and only if there exist an i such that $C^i(w) \approx w'$. It is obvious that cyclic permutability forms an equivalence relation for any group G.

Proposition 2.3.4 *For any group G, the equivalence relation of cyclic permutability ($\approx_{cp}$) is identical to that of conjugacy ($\approx_c$).*

Proof. Any group G has a presentation which may be obtained from some free group $\mathcal{F}_n$ of rank n by adding relations [51]. Moreover, if the conjugacy problem is solvable in one representation, it is solvable in all [110]. Suppose $w \approx_{cp} w'$, then there exists an i for which

$$w' \approx C^i(w) = f_{s_j}^{p'_j} \cdots f_{s_{m-1}}^{p_{m-1}} f_{s_m}^{p_m} f_{s_1}^{p_1} f_{s_2}^{p_2} \cdots f_{s_j}^{p''_j} \tag{2.93}$$

where

$$p'_j + \sum_{k=j+1}^{m} p_k = i \tag{2.94}$$

Let

$$\gamma = f_{s_1}^{p_1} f_{s_2}^{p_2} \cdots f_{s_j}^{p''_j} \tag{2.95}$$

Then

$$\begin{aligned} w' &\approx f_{s_j}^{p'_j} \cdots f_{s_{m-1}}^{p_{m-1}} f_{s_m}^{p_m} \gamma && (2.96)\\ &\approx \gamma^{-1}\gamma f_{s_j}^{p'_j} \cdots f_{s_{m-1}}^{p_{m-1}} f_{s_m}^{p_m} \gamma && (2.97)\\ &\approx \gamma^{-1} w \gamma && (2.98) \end{aligned}$$

Thus we have $w \approx_c w'$. Now suppose $w \approx_c w'$, then there exists a γ such that

$$w' \approx \gamma^{-1} w \gamma \tag{2.99}$$

If the word length of γ is $L(\gamma)$, then we have

$$C^{L(\gamma)}(w') \approx \gamma\gamma^{-1} w \approx w \tag{2.100}$$

Thus $w \approx_{cp} w'$. □

We will refer to the set of words which contains the word w and all its cyclic permutations as the *cyclic word* $c(w)$. If $L(w) = m$, then this set contains $|c(w)| = m$ elements. Given two cyclic words $c(w)$ and $c(w')$ we test their equivalence by attempting to construct an isomorphism $\iota : c(w) \to c(w')$ such that $\iota(a) = a$ for all $a \in c(w)$. Clearly $|c(w)| = |c(w')|$ is a necessary condition for the existence of ι. If and only if ι exists, the cyclic words are considered equal, $c(w) = c(w')$. If and only if $c(w) = c(w')$, we have $w \approx_c w'$ by proposition 2.3.4. The set $c(w)$ may be visualized as the word w "made circular" as in figure 2.4.

The existence of ι may be established by testing the members of $c(w)$ for equality with the members of $c(w')$ pairwise in the following manner: Select from $c(w)$ an arbitrary member, a say. Check a for equivalence with all members of $c(w')$. Clearly, if and only if there exists a $b \in c(w')$ such that $a = b$, an ι exists. Since every word has length m and there are m words in $c(w')$, this comparison will take a time proportional to m^2. Thus it is possible to test the equivalence of two cyclic words of length m with complexity $O\left(m^2\right)$.

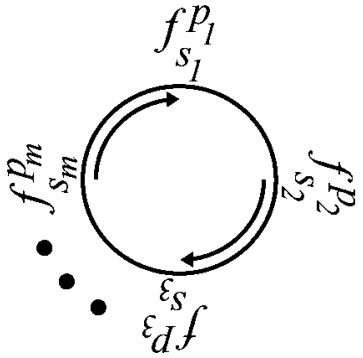


Figure 2.4: The word w given in equation (2.89) bent into a circle. While the circularity removes the notions of beginning and end of a word, it preserves the directionality of it.

Rewriting Systems for Cyclic Words

We shall call a TRS *cyclicly terminating*, *cyclicly confluent* and *cyclicly locally confluent* if it is respectively terminating, confluent and locally confluent under application to all cyclic words over the alphabet of the TRS. It is obvious from the above discussion that a cyclicly complete TRS solves the conjugacy problem. For this reason it is important to develop results about cyclic completeness along the lines of the results for linear words in order to obtain a conjugacy solution.

Termination in Cyclic Rewriting Systems

We have seen that a TRS $\mathcal{R}$ terminates if and only if a reduction order exists [9]. In what follows, we shall assume that this reduction order is a total order; note that this is a stricter requirement than that of a reduction order. Suppose that the alphabet of $\mathcal{R}$ is $\mathcal{A} = \{f_i\}$ for $1 \leq i \leq p$. By assumption, p is finite. Consider the total order $<_{\mathcal{R}}$ defined by $f_i <_{\mathcal{R}} f_{i+1}$ for all i. This can be done without loss of generality as a mapping from $\mathcal{A}$ to itself can change the order. Recall that $\mathcal{R}$ terminates if and only if $r_i <_{\mathcal{R}} l_i$ for every rule $l_i \to r_i \in \mathcal{R}$.

We introduce an integer valued *weight* metric function $g(w)$ and an integer valued *length* metric function $L(w)$ on the set of words w written on the alphabet $\mathcal{A}$. The metrics satisfy

$$\begin{aligned} g\left(f_{a_1} f_{a_2} \cdots f_{a_m}\right) &= g\left(f_{a_1}\right) + g\left(f_{a_2}\right) + \cdots + g\left(f_{a_m}\right) && (2.101)\\ L\left(f_{a_1} f_{a_2} \cdots f_{a_m}\right) &= L\left(f_{a_1}\right) + L\left(f_{a_2}\right) + \cdots + L\left(f_{a_m}\right) && (2.102)\\ L\left(f_i\right) &= 1 && (2.103)\\ g\left(f_i\right) &< g\left(f_{i+1}\right) && (2.104) \end{aligned}$$

We shall call a rule *length reducing* if $L(r_i) < L(l_i)$ and *weight reducing* if $g(r_i) < g(l_i)$. Any rule is a *c-obstruction* (for commutation-obstruction) if and only if it keeps constant both length and weight. That is, it is a rule which changes the position of the letters only.

A c-obstruction obstructs cyclic termination as there exist cyclic words which would give rise to an infinite rewriting chain due the changing of relative position of subwords by the c-obstruction. An example is the cyclic word $c(\alpha\beta\alpha\beta)$ under the TRS $\mathcal{R} = \{\alpha\beta \to \beta\alpha\}$. The rewriting chain will loop between the two states $c(\alpha\beta\alpha\beta)$ and $c(\beta\alpha\alpha\beta)$; the period of the loop may, in general, be arbitrarily large. Such looping may be dealt with in two ways. Firstly, one may compare each new cyclic word with the entirety of the rewrite chain so far enumerated. If equality is found, looping has been detected and one may stop. Secondly, one may determine if a subword of the current word commutes with the rest of the word. If this can be determined and such a subword is found, the subword may be extracted from the word and the two words should then be rewritten separately. The first method is computationally expensive and does not produce a unique normal form as we would have to consider the entire loop at the end of the rewrite chain as the identifying set of the word. The second method is not necessarily applicable but if it is, it will terminate in a set of subwords which uniquely identify the word. The advantage of the second method over the first is that the number of elements in the set has an upper bound.

We conjecture that a TRS $\mathcal{R}$ cyclicly terminates if and only if it terminates and contains no c-obstructions or contains c-obstructions that can be removed in the above way.

Confluence in Cyclic Rewriting Systems

Newman's Lemma [120] extends easily to the cyclic case as we show below.

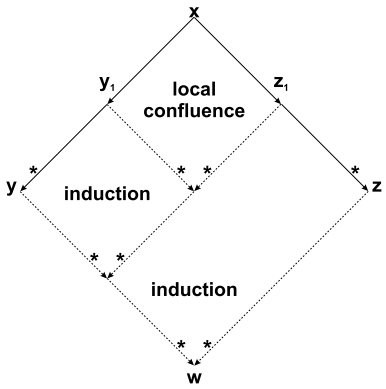


Figure 2.5: The proof of Newman's Lemma (lemma 2.3.5) in diagrammatic form. We begin at the top with a local divergence which is rectifiable by assumption and thus by induction any global divergence is also rectifiable. It is because of this diagrammatic proof that Newman's Lemma is also known as the Diamond Lemma.

Lemma 2.3.5 *A cyclicly terminating TRS $\mathcal{R}$ is cyclicly confluent if and only if it is cyclicly locally confluent.*

Proof. This proof is similar to the one given for the standard Newman Lemma in [80]. The result is obvious from figure 2.5.

(if) We want to show that if $y \leftarrow^* x \rightarrow^* z$, then the final forms of y and z are identical, which exist since $\mathcal{R}$ cyclicly terminates. If $x = y$ or if $x = z$, the result is obvious. If $x \rightarrow y_1 \rightarrow^* y$ and $x \rightarrow z_1 \rightarrow^* z$, then there exists a u such that $y_1 \rightarrow^* u \leftarrow^* z_1$ by cyclic local confluence. The existence of a w such that $y \rightarrow^* w \leftarrow^* z$ follows by induction over arbitrary length rewriting paths; the finiteness of all rewriting paths is attested to by cyclic termination.

(only if) This it trivial as cyclic local confluence is subsumed by cyclic confluence. □

The Critical Pair Lemma states that a TRS is locally confluent if and only if it has no critical pairs. Recall that a critical pair arises from an overlap of two redexes in a word which gives rise to a local divergence of rewriting paths which do not meet again. Given a TRS $\mathcal{R} = \{(l_i, r_i)\}$, a *cyclic overlap* is a cyclic word $c(w) = c(abcd)$ such that $abc = \rho l_i$ and $cda = \eta l_j$ for some words a, b, c and d, two (possibly equal) integers i and j and substitutions ρ and η. The cyclic overlap $c(abcd)$ is rewritten to both $c(\rho r_i d)$ and $c(b\eta r_j)$. A cyclic overlap is *non-critical* if the reducts are joinable, $c(\rho r_i d) \leftrightarrow^\star_{\mathcal{R}} c(b\eta r_j)$ and *critical* otherwise. A *cyclic critical pair* is the (unordered) pair of cyclic words $(c(\rho r_i d), c(b\eta r_j))$ which arises from a cyclic critical overlap. It is obvious that if $\mathcal{R}$ contains cyclic critical pairs, it can not be cyclicly confluent.

For example, consider the rewrite system $\mathcal{R} = \{abxba \rightarrow cxc\}$ over the alphabet $\mathcal{A} = \{a, b, c\}$ and some variables x and y. Clearly $\mathcal{R}$ contains the cyclic critical overlap *abxbabyb* which is to be rewritten into *bxbcyc* and *cxcbyb*. This cyclic critical pair may be resolved by noting that if the variable contained between the c letters is less than the other, it is that cyclic word which is to be prefered under the lexicographic order $c < b < a$. That is, we have to add a conditional rule depending on the relative value of the variables. This global rule must be applied, if applicable, with preference over the ordinary local rule. In this way we have extended Knuth-Bendix completion to the cyclic case; note that all rules added in this procedure are *global* whereas the usual rules of normal TRS's are *local*. We shall now prove the extention of the Critical Pair Lemma for the cyclic case.

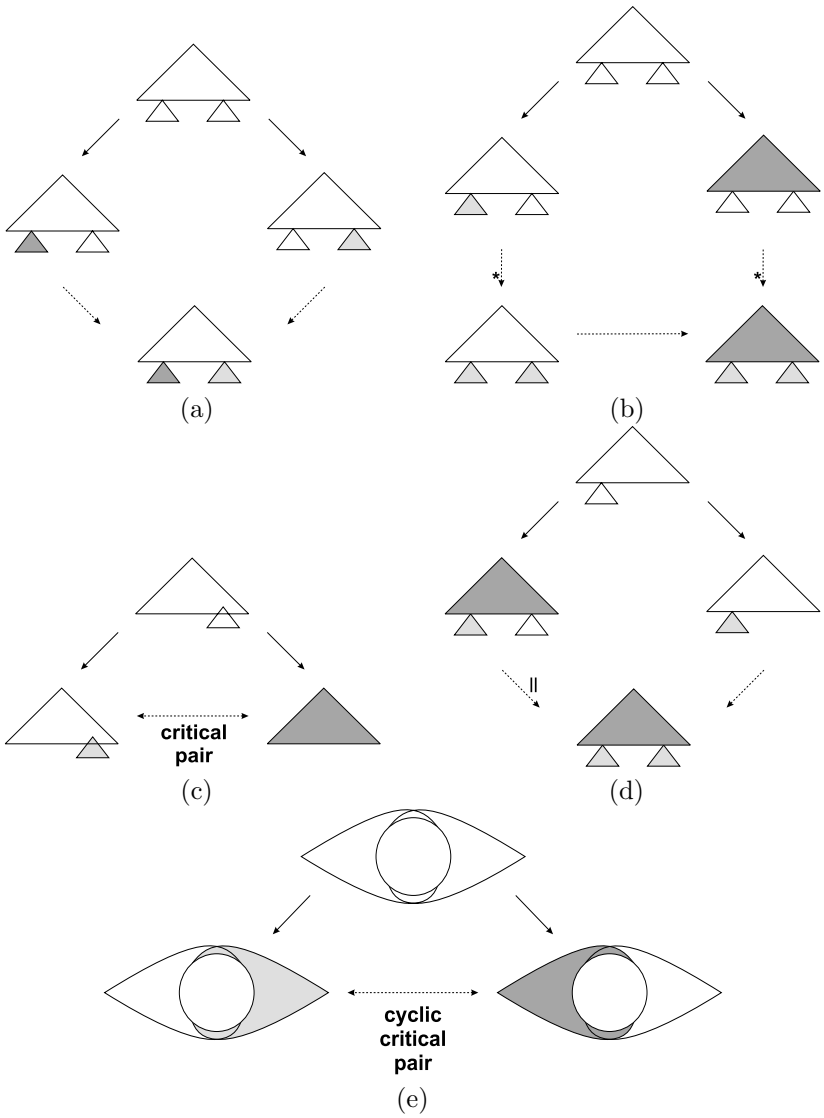


Figure 2.6: The proof of lemma 2.3.6 in its four cases: (a) the disjoint case, (b) the variable overlap case, (c) the critical overlap case, (d) the orthogonal case, (e) the circular critical overlap case.

Lemma 2.3.6 *A TRS $\mathcal{R} = \{(l_i, r_i)\}$ is cyclicly locally confluent if and only if it contains neither critical nor cyclicly critical pairs.*

Proof. We consider all relative positions of two redexes l_i and l_j in a cyclic word w and analyze them in turn. The first four cases occur in the standard Critical Pair Lemma but in the cyclic case there are five cases:

1. *(disjoint)* Suppose $c(w) = c(l_i x l_j y)$ for some words x and y. The existence of the common reduct $c(r_i x r_j y)$ is obvious; see figure 2.6 (a).
2. *(variable overlap)* Suppose l_i contains a variable which contains l_j as a subterm. If $l_j \to r_j$ does not change the applicability of l_i, a common reduct is obvious. If it does and the divergence does not resolve, we have an instance of a critical pair; see figure 2.6 (b).
3. *(critical overlap)* Suppose l_i and l_j have a critical overlap. A critical pair exists and obviously prevent local confluence; see figure 2.6 (c).
4. *(orthogonal)* Suppose l_i and l_j have a non-critical overlap. By definition the divergence resolves; see figure 2.6 (d).

5. *(cyclical critical overlap)* Suppose l_i and l_j have a cyclic overlap. If it is critical, we have an instance of a cyclic critical pair which obviously prevents cyclic local confluence. If it is non-critical, a common reduct exists by definition; see figure 2.6 (e).

□

It should be emphasized that the proof lemma 2.3.6 does not make any assumptions about the termination of $\mathcal{R}$. So we have a definite method for attempting to find a conjugacy problem solution in terms of rewriting. We shall use the braid groups to give an example of this completion process.

Yoder's Theorem

An alternative way to conceive of a conjugacy problem solution given a word problem solution was originally reported in Margaret Yoder's PhD thesis [161] and published in [126]; our presentation will deviate from both slightly.

Theorem 2.3.7 *(Pedersen and Yoder [126]) Let $G = \langle \{g_1, g_2, \cdots, g_i\}; S\rangle$ be a finitely presented group, let $\mathcal{A} = \{g_j^{\pm 1}\}$ for $1 \leq j \leq i$ and let $\mathcal{A}^*$ denote the set of all words constructible on the elements of $\mathcal{A}$. Further, let*

$$C = S \cup \{g_i^{\pm 1} g_i^{\mp 1} = e; s g_i^{\pm 1} x g_i^{\mp 1} f = sxf : \ x \in \mathcal{A}^*\} \tag{2.105}$$

and let $\mathcal{T}$ be a complete TRS for the monoid $M_C(G) = \langle \mathcal{A} \cup \{s, f\}; C\rangle$. Words u and v are conjugate in G if and only if suf and svf have identical final forms under $\mathcal{T}$.

Proof. *(if)* Suppose that suf and svf have identical final forms under $\mathcal{T}$. Since $u, v \in \mathcal{A}^*$, they do not contain s or f. Suppose that $suf \rightarrow_{\mathcal{T}}^* w$, then w is equivalent to suf in $M_C(G)$ since $\mathcal{T}$ is complete. Since no relation in $M_C(G)$ allows the positions of s or f to be changed, w is of the form $w = sw'f$ and by equivalence we have $suf \approx_{M_C} sw'f$. Since s and f are stationary symbols and u does not contain them, neither does w' and we have $u \approx_{M_C} w'$ in $M_C(G)$, thus u and w' are conjugate in G. Since we have assumed that suf and svf have identical final forms under $\mathcal{T}$, we have $u \approx_{M_C} w'$ and $v \approx_{M_C} w'$ and since conjugacy is an equivalence relation, we have $u \approx_{M_C} v$.

(only if) Suppose now that u and v are conjugate in G. Then there exists a word $w \in G$, such that $u \approx_G wvw^{-1}$. Since $w \in G$, we have that $w \in \mathcal{A}^*$ and thus the relation $swvw^{-1}f \approx_{M_C} svf$ is contained in $M_C(G)$. Since $u \approx_G wvw^{-1}$, we have that $suf \approx_{M_C} swvw^{-1}f \approx_{M_C} svf$. Since two conjugate words in G are equivalent in the monoid $M_C(G)$ and $\mathcal{T}$ is complete, the final forms of suf and svf are identical. □

Note that theorem 2.3.7 only asserts the existence of a conjugacy solution if the complete rewriting system $\mathcal{T}$ can be found. The new generators s and f are called markers because they mark the start and finish of a word. They are never contained in the middle of any word and no relation ever moves them from their positions. Their function is a notational convenience with which we may easily write down a relation which applies to a whole word and not a proper subword. The second relation which we have added to S is an example of this.

A conjugacy problem solution can obtained if Knuth-Bendix completion works for the new monoid. For our present situation, this approach does not, in fact, yield a complete TRS. Below we shall develop our own method of extending our word problem solution.

2.3.3 The Conjugacy Problem in B_n

Consider the TRS $\mathcal{W}_n$ of equation 2.87. We have already shown that $\mathcal{W}_n$ is complete and solves the word problem for B_n. We shall find that it is neither cyclicly terminating nor cyclicly locally confluent. By a Knuth-Bendix-like completion process, we are able to obtain a system which is cyclicly locally confluent. This system will be cyclicly terminating if all c-obstructions are removed, which is possible. The result will be a cyclicly complete system which thus solves the conjugacy problem in B_n.

Table 2.2: Cyclic overlaps between rules in $\mathcal{W}_n$.

$\rightarrow$	Cyclic Overlap	Final Forms
2, 2	$\prod_{i=1}^{n-1}(d_{i,1}a_{1,i}S_i)$	$\prod_{i=1}^{n-1} S_i$
	& $S_{n-1} = S'_{n-1}\sigma_k$; $3 \leq k < n$	$\sigma_1\sigma_k\sigma_1 S_1 \prod_{i=2}^{n-2}(d_{i,1}a_{1,i}S_i)\, d_{n-1,1}a_{1,n-1}S'_{n-1}$
3, 3	$\sigma_i\sigma_{i-1}P\sigma_i\sigma_{i-1}P'$	$\sigma_{i-1}\sigma_i\sigma_{i-1}P\sigma_{i-1}P'$
		$\sigma_{i-1}\sigma_i\sigma_{i-1}P'\sigma_{i-1}P$
3, 4	$\sigma_i\sigma_{i-1}Q\sigma_{i-1}R\sigma_i\sigma_{i-1}P$	$\sigma_{i-1}Q\sigma_{i-1}R\sigma_{i-1}\sigma_i\sigma_{i-1}P$
	& $P = d_{i-2,j}P'$	$\sigma_{i-1}\sigma_i\sigma_{i-1}Qd_{i-1,j}\sigma_i R^+P'$
4, 4	$\sigma_i\sigma_{i-1}Q_1\sigma_{i-1}R_1\sigma_i\sigma_{i-1}Q_2\sigma_{i-1}R_2$	$\sigma_{i-1}\sigma_i\sigma_{i-1}Q_1d_{i-1,j}\sigma_i R_1^+Q'_2\sigma_{i-1}R_2$
	& $Q_1 = d_{i-2,j}Q'_1$; $Q_2 = d_{i-2,j}Q'_2$	$\sigma_{i-1}\sigma_i\sigma_{i-1}Q_2d_{i-1,j}\sigma_i R_2^+Q'_1\sigma_{i-1}R_1$

In table 2.2, we list all four cyclic overlaps between the rules of $\mathcal{W}_n$ and the two final forms per overlap depending on the chosen rewrite path. Note that all overlaps are critical and that the cyclic overlap refers to the entire cyclic word.

Consider the TRS $\mathcal{G}_n$ below which is understood to contain only global rules for cyclic words, i.e. the entire word has to be matched to redexes in $\mathcal{G}_n$. The restrictions on the variables are identical to those of $\mathcal{W}_n$. The ordering $<_s$ is the standard shortlex ordering, i.e. words are sorted lengthwise first and then lexicographically using $<_b$.

$$\begin{aligned}\mathcal{G}_n = \{\,&(1)\ \prod_{i=1}^{n-1}(d_{i,1}a_{1,i}S_i) \rightarrow \prod_{i=1}^{n-1} S_i;\\ &(2)\ \sigma_i\sigma_{i-1}P\sigma_i\sigma_{i-1}P' \rightarrow \sigma_{i-1}\sigma_i\sigma_{i-1}P\sigma_{i-1}P' \text{ if } P <_b P'\\ &\text{ or } \sigma_{i-1}\sigma_i\sigma_{i-1}P'\sigma_{i-1}P \text{ if } P' \leq_b P;\\ &(3)\ \sigma_i\sigma_{i-1}Q\sigma_{i-1}R\sigma_i\sigma_{i-1}P \rightarrow \sigma_{i-1}Q\sigma_{i-1}R\sigma_{i-1}\sigma_i\sigma_{i-1}P;\\ &(4)\ \sigma_i\sigma_{i-1}Q_1\sigma_{i-1}R_1\sigma_i\sigma_{i-1}Q_2\sigma_{i-1}R_2 \rightarrow\\ &\sigma_{i-1}\sigma_i\sigma_{i-1}Q_1d_{i-1,j}\sigma_i R_1^+Q'_2\sigma_{i-1}R_2 \text{ if } R_1 <_s R_2\\ &\text{ or } \sigma_{i-1}\sigma_i\sigma_{i-1}Q_2d_{i-1,j}\sigma_i R_2^+Q'_1\sigma_{i-1}R_1 \text{ if } R_2 \leq_s R_1\ \}\end{aligned} \tag{2.106}$$

As described in the solution to the word problem, we will regard a cyclic word $c(w)$ as a pair $c(w) = (k, c(w_s))$. The first entry is an even integer counting the number of copies of Δ_n^2 in $c(w)$. The second entry is the rest of the word written in the σ_i. We shall now present an algorithm for the conjugacy problem in terms of $\mathcal{W}_n$ and $\mathcal{G}_n$. We prove, in a set of lemmas, that this algorithm solves the conjugacy problem in B_n.

Algorithm 2.3.8 *Input: A cyclic braid word $c(w)$. Output: A set of cyclic braid words which collectively are a unique representative of the conjugacy class of w.*

1. Apply rule 1 of $\mathcal{W}_n$ as many times as possible.
2. Test if w is splittable, i.e. if it is in the form $w = w_1w_2$ where w_1 commutes with w_2. If it is, separate w_1 and w_2 and treat them separately from now on. If not, do nothing. Note that we are testing the linear word w and not $c(w)$.
3. If applicable, apply any rule in $\mathcal{G}_n$ and proceed with step 5. If not continue with the next step.
4. Apply any of rules 2 to 4, in that order of priority, of $\mathcal{W}_n$ exactly once to each of the separated cyclic braid words, if possible.
5. Go back to step 2 of the algorithm and continue until there is not braid word which may be split further and no braid word to which any of the rules in $\mathcal{W}_n$ and $\mathcal{G}_n$ are applicable.

6. The number k and the set of split braid words are now collectively the unique representative required.

We note that because of the restrictions on the variable S_{n-1}, rule 5 of $\mathcal{W}_n$ and rule 1 of $\mathcal{G}_n$ are identical. It is obvious from the algorithm that if it cyclicly terminates and $\mathcal{W}_n \cup \mathcal{G}_n$ is cyclicly confluent, then the conjugacy problem in B_n is solved by it.

Lemma 2.3.9 *Algorithm 2.3.8 cyclicly terminates in a time $O\left(l^5 n^{11}\right)$, where l is the initial word length.*

Proof. Suppose initially that the word length of $w \in B_n$ is l and w contains exactly m inverse generators. Step 1 of the algorithm applies rule 1 of $\mathcal{W}_n$ as many times as possible, which is clearly m times. After such replacement, the word length of w is now

$$L(w) = l + mn(n-1) - m \tag{2.107}$$

Since all inverse generators are now gone and no rule creates further inverse generators, rule 1 can never be applied again. Note also that no other rule increases the length of the braid word. Such replacements may be made in $O\left(ln^2\right)$. Steps 2 to 4 of the algorithm are looped now. Since the total length of the braid words is bounded by equation 2.107, a split may occur only a finite number of times.

We test if the word w is splittable. There are exactly $L(w)$ words w_1 and w_2 such that $w_1 w_2 = w$ and we must test if $w_1 w_2 \approx w_2 w_1$ which is a word problem which can be solved in $O(L(w)^2 n^4)$ according to theorem 2.3.3. In fact there is a solution by Birman [22] with complexity $O(L(w)^2 n)$ which is the complexity we shall assume holds here. Thus testing splittability may be done in at worst $O\left(L(w)^3 n\right)$.

Rule 1 of $\mathcal{G}_n$ reduces the total length while no rule increases it and thus it may be applied only a finite number of times. By the same analysis as for the word case, it is bounded by $O(l)$. Rule 2 of $\mathcal{W}_n$ is the commutation relation, the possibility of the infinite application of which is explicitly removed in step 2 of the algorithm. It is at worst a comparison between every generator and so bounded by $O\left(L(w)^2\right)$. Taken independently, the other rules also terminate as rule 3 of $\mathcal{W}_n$ and rules 2 and 3 of $\mathcal{G}_n$ decrease total generator index count and rule 4 of $\mathcal{W}_n$ and rule 4 of $\mathcal{G}_n$ increase it. Since total generator index count is bounded from below by $L(w)$ and above by $(n-1)L(w)$ the application of these rules must terminate independently. We must show that there can be no interference between the rules which would give rise to infinite rewriting. Rules 2, 3 and 4 of $\mathcal{G}_n$ require that the entire word contains exactly two σ_i. If rule 4 can be applied, this number may increase or stay equal to two. If it increases, the only way to decrease it is to use rule 3 of $\mathcal{W}_n$. It is obvious from the rules that this process will never lead to the original cyclic word again even though the number of σ_i may again reach two.

If rule 4 of $\mathcal{W}_n$ is used to raise the number of σ_i, it may be lowered by using rule 3 of $\mathcal{W}_n$ or rules 2 or 3 of $\mathcal{G}_n$. This process can also never again reach the original word as a local ordering in the form $\sigma_{i-1}\sigma_i\sigma_{i-1}$ is formed and never undone. The number of applications of all these rules is clearly bounded by $O(L(w))$. As the number of times the loop is to be performed is of order $L(w)^2$ and the worst-case step inside the loop is of order $L(w)^3 n$, the whole algorithm runs in $O\left(L(w)^5 n\right) = O\left(l^5 n^{11}\right)$. □

Lemma 2.3.10 *Algorithm 2.3.8 is cyclicly confluent.*

Proof. By theorem 2.3.1, $\mathcal{W}_n$ is confluent and thus contains no critical pairs. Algorithm 2.3.8 uses $\mathcal{G}_n$ as well as $\mathcal{W}_n$. By construction, $\mathcal{G}_n$ resolves all the cyclic critical pairs of $\mathcal{W}_n$ but, as may be easily verified, introduces no further critical pairs or cyclic critical pairs. By lemma 2.3.6 this is a necessary and sufficient condition for cyclic local confluence. By lemma 2.3.9, the algorithm cyclicly terminates and so, by lemma 2.3.5, the algorithm is confluent. □

As the algorithm terminates and is confluent, it solves the conjugacy problem in B_n with computational complexity $O\left(l^5 n^{11}\right)$.

Chapter 3

The Minimum Word Problem

A well-known problem of combinatorial braid theory is the minimization problem: Given a braid $A \in B_n$ find a braid A_m such that $A \approx A_m$ and $L(A_m) \leq L(A^*)$ for any braid $A^* \approx A$ where $L(A)$ denotes the word length of the braid A. In this chapter, we prove that this problem is NP-Complete and we find an algorithm for it.

3.1 Introduction

In the Artin representation of B_n, the number of generators required to write down a braid word, its length, is equal to the crossing number of the topological braid. In practice, we find that by moving a few of the strings of the topological braid, its crossing number may be reduced, making the braid simpler. It would be especially useful to possess a general method to compute an equivalent braid of minimum crossing number. Apart from many applications, this problem is well-known in combinatorial braid theory and is of independent mathematical interest.

Given a braid $A \in B_n$ in the Artin generators, the question whether there exists an equivalent braid $A' \in B_n$ of shorter length has been shown to be NP-Complete by Paterson and Razborov [125]. Not only does this mean that this question is computationally equivalent to all other NP-Complete problems, it also means that (unless P = NP, the meaning of which will become apparent in our review of NP-Completeness) any algorithm which answers the question would execute in exponential-time in n. Since Paterson and Razborov's result refers to the minimization problem for general n, we ask whether it is also an NP-complete question for particular n. This question is explicitly asked as open question 9.5.6 on page 209 of [59] and it seems to have been negatively answered in an unpublished preprint by Tatsuoka five years earlier but we were unable to obtain it [146].

In proving the NP-Completeness of the problem, Paterson and Razborov showed that the problem can be reduced to a known NP-Complete problem. This does not however provide a usable algorithm. For 3-braids, a linear complexity algorithm has been found [17] but no general algorithm for $n > 3$ exists. A minimization algorithm in the band-generator presentation of the braids groups has been found for $n = 3, 4$ but the length of the braid in this presentation is not equal to the crossing number [159] [89]. It is untypical of a group for which the word problem is solvable that no unique normal form of minimal length in some naturally arising presentation exists for the braid groups. A unique normal form of minimal length in certain natural presentations of free groups, HNN-extensions and free products exists, for example.

After a little experimentation, it is clear that a braid must, in general, be increased in length before it may be reduced to minimum length algebraically. An example of such a braid is given in [17]. We show that a certain readily obtained braid provides an upper bound for this necessary increase in length and prove several properties of this braid. We explicitly construct a set of words which must be searched for a certain property in order to obtain a minimal length representative of any braid. This constitutes an algorithm to solve the minimization problem. Since the set of words which must be searched is, in the worst case, exponential in size, the algorithm takes an exponential amount of time to complete.

3.2 NP-Completeness

First, we review the basic ideas and results of the theory of NP-Completeness and then we shall prove that a particular problem, known as SORTING DOES NOT MINIMALLY PARTITION, is NP-Complete.

3.2.1 A Review of NP-Completeness

In this section, we shall provide a very concise review of the main results of the theory of NP-Completeness (NPC) and complexity in general. For reasons of space, this review will be informal. The full details and all the proofs for the statements we make herein are in [63].

When concerned with practically solving combinatorial problems, we often wish to use mechanical aid and the question of resources arises. It is, in general, straightforward to give a natural measure of the input size $\mathcal{S}$ of a problem Π (any graph problem input is usually measured by the number of edges and vertices in the input graph, for example) and we analyze an algorithm which solves our combinatorial problem in order to obtain a function $f(\mathcal{S})$ which gives the *maximum* amount of computing time required to solve a problem of size $\mathcal{S}$ (thus $f(\mathcal{S})$ is a worst-case measure of time). By the *(worst-case) complexity* $\mathcal{C}(\Pi)$ of the problem Π we mean the asymptotic behavior of $f(\mathcal{S})$ as $\mathcal{S} \to \infty$. An algorithm is *linear* when $f(\mathcal{S}) \to \mathcal{S}$ and *quadratic* when $f(\mathcal{S}) \to \mathcal{S}^2$ as $\mathcal{S} \to \infty$. We consider an algorithm to be *efficient* if and only if $\mathcal{C}(\Pi)$ is bounded from above by a polynomial function of $\mathcal{S}$ and *inefficient* or *intractable* otherwise. Apart from desiring as quick an algorithm as possible for Π, we wish to know in general whether an efficient algorithm exists.

It is customary to refer to the totality of information necessary to be specified for a particular problem Π before it can be solved as an *instance* of Π. A *decision problem* Π (intuitively a question with "yes" or "no" answers) consists of two sets D_Π of possible instances and $Y_\Pi \subseteq D_\Pi$ of yes-instances. The problem consists of deciding whether the particular instance specified lies in Y_Π. The theory of NPC deals primarily with decision problems but may be extended to more general types of problems.

Computation may be modeled in a variety of ways all of which can be simulated in terms of each other; the most frequently used model is the Turing machine, which we shall use here. A *deterministic Turing machine* (DTM) is a machine which possess a finite-state control, a read-write control and an infinite length of tape (there exist models with more than one tape but we shall not need them). The finite-state control will tell the machine what to read or write using the read-write head on the infinite tape on which it does all its work. A *program* for a DTM consists of an alphabet of symbols to be used for reading and writing (and specifying the input), a finite set of states for the DTM to be in and a transition function. The states include two distinguished states $\mathcal{T}_y$ and $\mathcal{T}_n$ and the program will halt if and only if either of these is reached; $\mathcal{T}_y$ will be the "yes" and $\mathcal{T}_n$ the "no" state which provide the answer to the problem Π. The transition function prescribes a new state for each present state as a function of the possible input. Thus a DTM supplied with a program and some input will traverse its states in accordance with the transition function and its input and will, in some cases, reach either $\mathcal{T}_y$ or $\mathcal{T}_n$. The question whether it will ever reach either $\mathcal{T}_y$ or $\mathcal{T}_n$ is known as Turing's Halting Problem and can not, in general, be answered for all inputs.

The number of symbols on the tape which describes the input will be our abstract measure of input size $\mathcal{S}$ and the number of evaluations of the transition function, to leading order, as a function of $\mathcal{S}$ will constitute the complexity $\mathcal{C}(\Pi)$ of the program. If $\mathcal{C}(\Pi)$ is bounded from above by a polynomial function of $\mathcal{S}$, then the Π is said to have *polynomial-time complexity*. The complexity class P is defined as the set of all problems for which a DTM program with polynomial-time complexity exists.

It has been observed that verifying that a suggested solution is true is, in many cases, easier than finding the solution. This idea gives rise to the concept of a *Non-deterministic Turing Machine* (NDTM) which consists of a DTM plus a guessing control. The guessing control guesses a solution and the DTM checks it. A NDTM program is *polynomial-time* if the checking stage has polynomial complexity in $\mathcal{S}$. The complexity class NP is defined as the class of all problems for which a polynomial-time NDTM program exists. Clearly P $\subseteq$ NP. Whether or not this inclusion is strict has been the topic of considerable discussion. It is believed by most in the area that P $\subset$ NP, however no proof of this has been found and a convincing though informal argument based on

empirical evidence can be made for both positions.

We define a problem Π to be *polynomially transformable* into a problem Π' (denoted $\Pi \propto \Pi'$) if and only if there exists a function f which takes instances I of Π and returns instances $f(I)$ of Π' such that: (i) $f(I) \in Y_{\Pi'}$ if and only if $I \in Y_{\Pi}$ and (ii) there exists a polynomial-time DTM for f. Note that $\Pi \propto \Pi'$ does not necessarily imply $\Pi' \propto \Pi$. In fact, the relation $\propto$ defines a partial order on the elements of NP. Using this, we may define the class NPC to be the set of problems Π such that: (i) $\Pi \in$ NP and (ii) for all other problems $\Pi' \in$ NP, we have $\Pi' \propto \Pi$. Thus the NPC problems are the hardest in NP with respect to the partial order defined by polynomial transformability. By construction of the class NPC, all NPC problems are polynomially transformable to all other NPC problems. That means that a polynomial-time algorithm for one NPC problem would immediately result in polynomial-time algorithms for all NPC problems and by extention all NP problems. That is, it would prove that NP = P.

It is not immediately obvious that there are any NPC problems but Cook proved that a problem known as satisfiability (SAT) is in NPC. This result is important as we note that this means that we can prove that a problem $\Pi \in$ NPC if and only if $\Pi \in$ NP and some known NPC (such as SAT) problem transforms to Π. This is a favorite method to prove that a problem is NPC.

3.2.2 Practicality of the Theory of NP-Completeness

There has been considerable controversy over the practical implications of the statement "problem Π is NPC." While many firm believers in P $\neq$ NP insist that this means Π is intractable and should be regarded as practically unsolvable, many disagree with this view. If we are actually concerned about solving Π in practice, we wish to know how long the solution will take in terms of real time. In many situations we are concerned with only a few special cases or statistical results (i.e. the *average case*) and so worst-case complexity or general statements about intractability are not specific enough to tell us whether we can solve a particular instance of Π. The original NPC problem of satisfiability for example can be solved with an algorithm of exponential worst-case complexity which however runs very quickly for the average case. This seems to be true for many NPC problems and bounds on what "average" means and how quickly the algorithms should run can probably be obtained using the methods described for the traveling salesman problem in [73]. It should be noted that it is, in general, impossible to tell *a priori* whether a given instance is sufficiently average to be solved quickly.

In this vein, the theories of approximation and random algorithms have arisen. If the exact solution to our specific instance does in fact take too long, then we are perhaps satisfied with an answer which is *almost* a solution (an approximation algorithm) or with an answer which is likely to be a solution with a probability very close to one (a random algorithm). Furthermore, a proof that Π is NPC does not, in general, provide a usable algorithm for Π. So if we wish to actually solve Π we must look beyond the proof of its NP-Completeness and make use of problem-specific features.

We prove that Π is NPC by reduction from a known NPC problem Π'. Even though, we may have good approximation algorithms and knowledge of the average case for Π', this does not translate to Π, in general. Which instances are average appears to be problem specific; approximations and random methods appear to have to make use of special features of the specific problem. Moreover, even if the algorithm has exponential complexity, as long as the actual time taken is relatively small, we may still be able to use such an algorithm for relatively small instances without practical difficulties. All this hints that the theory of NP-Completeness may not provide a very useful guide for the practical solution of an NPC problem other than giving a convincing argument (as long as P $\neq$ NP is not proven or disproven) that the worst-case *necessarily* has exponential complexity.

Many problems of practical significance are in NPC. While most people assume that NP $\neq$ P, if a counterexample were ever found, the practical importance of that example would be great. It would, however not lead to an immediate revolution of practical computation because the tree of polynomial transformations between the NPC problems is so complex and frequently inefficient that this one example would not provide a practical polynomial-time algorithm to solve other NPC problems without considerable extra work. From a practical point of view, an algorithm which goes as n^{10} may not be any better than one which goes as 2^n, it all depends on the problem and the actual resources required and available.

3.2.3 Sorting Does Not Minimally Partition

Suppose we have a set $N_r = \{1, 2, \cdots, r\}$, then the set N_r^m is the set of all words on the letters N_r of length m and $N_r^* = \bigcup_m N_r^m$. We define the number of *inversions* $inv(q, \pi)$ of a word $q = q_1 q_2 \cdots q_m \in N_r^m$ with respect to a permutation π on N_r to be the number of adjacent letter interchanges necessary to sort the letters of the word q with respect to π. For example, suppose $r = 4$, $\pi = [2, 1, 4, 3]$ and $q = 432413$, then we want to sort q into $q' = 214433$ which requires $inv(q, \pi) = 6$ adjacent letter interchanges. Note that whatever π, $inv(q, \pi) \leq m(m-1)/2$. The identity permutation is denoted by ι.

SORTING DOES NOT MINIMALLY PARTITION (SNMP)
INSTANCE: $q \in N_r^*$.
QUESTION: Is there a permutation π of N_r such that $inv(q, \pi) < inv(q, \iota)$?

SNMP was first shown to be NPC by Paterson and Razborov [125] but we give a different proof here. As mentioned above, the book [63] is considered the standard reference on the theory of NPC. Amongst a solid review of the field, it contains a list of several hundred NPC problems. We shall consider the problems in that list well-known and will not give proof that they are NPC. The problem GROUPING BY SWAPPING (GBS) (problem SR21 in [63], p. 231) is one of these and we shall prove SNMP to be NPC by restricting GBS. (GBS is proved to be NPC by a transformation from FEEDBACK EDGE SET (GT8, p.192) → VERTEX COVER (GT1, p. 190) → 3SAT (LO2, p. 259) → SAT (LO1, p. 259) which is the original NPC problem. The references in parenthesis are to [63].) To be as clear as possible, we quote GBS exactly as it appears in [63], modulo notation.

GROUPING BY SWAPPING (GBS)
INSTANCE: Finite alphabet N_r, string $q \in N_r^*$, and a positive integer K.
QUESTION: Is there a sequence of K or fewer adjacent symbol interchanges that converts q into a string s in which all occurences of each symbol $a \in N_r$ are in a single block, i.e. s has no subsequences of the form aba for $a, b \in N_r$ and $a \neq b$?

Theorem 3.2.1 *SNMP is in NPC.*

Proof. Note that the string s which has no subsequences of the form aba for $a, b \in N_r$ such that $a \neq b$ is sorted with respect to some permutation π on N_r. Thus there exists a sequence of K or fewer adjacent symbol interchanges if and only if there exists a permutation π on N_r such that $inv(q, \pi) \leq K$. If we put $K = inv(q, \iota) - 1$, an instance of GBS becomes an instance of SNMP. This is known as a proof by restriction (see [63]) as we have shown that a specific type of instance of GBS is an instance of SNMP; it clearly follows that SNMP is in NPC since GBS is. □

3.3 Non-Minimal Braids is NP-Complete

In this section, we state formally the problem NON-MINIMAL BRAIDS (NMB) and show that it is NP-Complete. Paterson and Razborov first proved this result by a reduction of a subproblem of NMB to SNMP. We shall follow their basic ideas but will present a substantially different proof. We do not assume that the reader is familiar with their proof and our discussion is completely independent.

3.3.1 Statement of the Problem

NON-MINIMAL BRAIDS (NMB)
INSTANCE: A braid group B_n and a word $A \in B_n$.
QUESTION: Is there a word $A' \in B_n$ such that $A' \approx A$ and $L(A') < L(A)$?

Clearly NMB can be answered by an algorithm which finds a minimum length representative A_m of the equivalence class of A by comparing $L(A_m)$ and $L(A)$. We define this problem formally,

MINIMAL EQUIVALENT BRAID (MEB)
INSTANCE: A braid group B_n and a word $A \in B_n$.
SEARCH: Find a word $A_m \approx A$ such that $L(A_m) \leq L(A')$ for any $A' \in B_n$ such that $A' \approx A$.

The class of problems of which MEB is a member is that of *search problems*. Search problems Π differ from decision problems mainly in that they consist of a set D_Π of possible instances and a set S_Π of *solutions*. For each instance $I \in D_\Pi$, an algorithm which solves Π returns either "no" if $S_\Pi = \emptyset$ or some solution $s \in S_\Pi$. The notion of NP-Completeness can be extended to search problems (see [63], p. 110) and by this extension it is clear that the search problem MEB is in NPC if and only if the decision problem NMB is in NPC.

We shall show that for a certain class of braids, NMB is in NPC. First, we construct this class, then we prove some of its properties and finally show that the question NMB for a member of this class can be reduced to SNMP. This shows that NMB is in NPC because it is in NP and a subproblem is in NPC.

3.3.2 The Weft Braids

Consider the identity braid (no crossings) in the braid group B_{cm+1} where we have partitioned the $cm+1$ strings into $m+1$ categories. The first category is constituted only by the braid on the far left and each other category is made up of exactly the next c strings to the right of the last category. Choose an integer $r < cm+1$ and assign a label to the m categories with c strings from the set $N_r = \{1, 2, \cdots, r\}$. This labeling induces a string V on N_r^{cm}. We denote the number of times label i occurs within V by $\#(V, i)$. We shall refer to the string on the far left as the *weft* (the weft is the string which weaves between the threads stretched lengthwise in a loom while making fabric), the categories bearing label i as *i-cables* and the strings in a particular i-cable as *i-wires*. In what follows the cables will act as units; that is, there will be no crossings between wires making up a particular cable. Choose a permutation π on N_r and construct the braid V_π from V and π by bringing all the $\pi(1)$-cables to the left of all the other cables except the weft. Then do the same for the $\pi(2)$-cables and so on, so that the final braid will have its cable labels sorted according to π. The cable currently brought to the front shall under-cross any cable in its path if and only if its label is lower than that cable. This defines the braid V_π exactly. Given a labeling V, the number of Artin braids V_π this can generate is equal to the number of permutations on r symbols, $r!$. If $c = 1$, $m = 5$ and $r = 3$, the labeling $V = 21312$ can generate 6 Artin braids, two of which are $V_\iota = \sigma_2\sigma_4\sigma_3\sigma_5$ and $V_{[2,1,3]} = \sigma_5^{-1}\sigma_4\sigma_3^{-1}\sigma_5$. Note that the word length $L(V_\pi)$ of the Artin braid V_π is $L(V_\pi) = inv(V, \pi)c^2$.

In addition to the choices made above, choose two further integer parameters t, s. A *weft braid* $U_\pi \in W_n(r, t, s, c)$ on n strings with parameters r, t, s, c induced by a permutation π on N_r is constructed by first constructing V_π for some labeling V as described above. Then the weft over-crosses all cables until it is just to the right of all the $\pi(r)$-cables, it then passes underneath all the $\pi(r)$-cables (which are now in a block) and finally passes over all of them and under all of them a *further* $t-1$ times (the weft has encircled the $\pi(r)$ block of cables t times). The weft then continues to the $\pi(r-1)$-cables and so on to do the same under-crossing cables if it is going to the left and over-crossing otherwise. The entire process of the weft encircling the blocks is repeated s times and U_π terminates in V_π^{-1}. We refer to braids in the above form as being in *weft form*.

Next, we prove three lemmas which will be used to show that finding minimal weft braids is in NPC. The first lemma was assumed by Paterson and Razborov as obvious, we shall prove it here. The other two lemmas are proven differently here than they were by Paterson and Razborov [125].

Lemma 3.3.1 *$U_\pi \approx U_{\pi'}$ for any two weft braids $U_\pi, U_{\pi'} \in W_n(r, t, s, c)$ defined by the same labeling V.*

Proof. Given the set of weft braids $W_n(r, t, s, c)$, choose a word $V \in N_r^{cm}$ where $m = (n-1)/c$ and two permutations π and π' on N_r. Construct the two weft braids in weft form U_π and $U_{\pi'}$ defined by the labeling V and the permutations π and π' respectively. We wish to show that $U_\pi \approx U_{\pi'}$. Note that we may set $\pi' = \iota$ without loss of generality.

Consider U_ι and focus attention upon the (adjacent) collections of i and $(i+1)$-cables at the beginning of the center braid, i.e. just after V_ι. Suppose we were to interchange their positions by pulling the i-cables under the $(i+1)$-cables while keeping the rest of the braid stationary. This

generates a number of new crossings of the form WW^{-1}. Move the second half of these crossings, W^{-1}, to the end of the center braid in order to conform with the definition of the weft form then the whole braid will be $V_\iota WCW^{-1}V_\iota^{-1}$ where C is the center braid. This motion will clearly alter the braid into exactly that form in which it would have been if the permutation had not been $\iota = [1, 2, 3, \cdots, r]$ but $[1, 2, \cdots, i-1, i+1, i, i+2, \cdots, r]$. In the same way, we may switch the order of any two adjacent pairs of labels in the permutation. Since any permutation π on N_r can be generated by a finite number of adjacent symbol interchanges in the identity permutation ι on N_r, we can generate all the weft braids in weft form (defined on the same initial labeling V) U_π from U_ι by braid group motions which proves the lemma. □

As U_π is independent of π, we shall omit the permutation π when no confusion can arise.

Lemma 3.3.2 *(Paterson and Razborov [125]) The length of a weft braid in weft form $U_\pi \in W_n(r, t, s, c)$ with $n = 1 + cm$ defined by the labeling V satisfies*

$$2inv(V, \pi)c^2 + 2tmcs \leq L(U_\pi) \leq 2inv(V, \pi)c^2 + (r + 1 + 2t)mcs \tag{3.1}$$

and in particular $L(U_\iota) = 2inv(V, \iota)c^2 + 2tmcs$.

Proof. By construction $L(V_\pi) = L(V_\pi^{-1}) = inv(V, \pi)c^2$. A single coil around the strings of label i has $2c\#(V, i)$ crossings. If $\pi = \iota$, then the weft over-crosses all cables first using cm crossings. It then under-crosses $c\#(V, r)$ times and coils $t - 1$ times using $2c(t-1)\#(V, r)$ crossings. The $(r-1)$-cables are just beside the r-cables and so no further crossings are needed, we under-cross them and coil again $t - 1$ times. So in the case that $\pi = \iota$, we use exactly

$$L(U_\iota) = 2inv(V, \iota)c^2 + 2tmcs \tag{3.2}$$

crossings. For the case $\pi = \iota$, the non-coiling crossings of the weft with the wires can be viewed as exactly one more coiling with each of the blocks of wires since this would give exactly the remaining number of crossings and geometrically looks like it. By construction of the general case, we require at least $2tmcs$ crossings for the center part of the braid for any other permutation π, this gives us the lower limit.

For $\pi \neq \iota$ we require these t coils plus a number of crossings of the weft and the wires in order to sort them in the order prescribed in the construction above. There are r different blocks of cables and m cables in total. Thus the maximum number of crossings of the weft with a cable in one stage of the construction is clearly $(r + 1)m$ but there are c wires per cable and s identical construction stages and so the total number is $(r + 1)mcs$. This must be added to the minimum number of required crossings and gives us the upper limit in the lemma. □

Lemma 3.3.3 *(Paterson and Razborov [125]) If $U \in W_n(r, t, s, c)$ obeys $L(U) \leq L(U')$ for any $U' \in W_n(r, t, s, c)$ such that $U \approx U'$, then*

$$L(U) \geq 2inv(V, \pi)c^2 + 2tmcs \tag{3.3}$$

for some permutation π.

Proof. Construct the weft braid U_π for some permutation π. It has $2inv(V, \pi)c^2$ crossings between wires of different labels. Wires of the same label never cross. Half of these crossings V_π are the inverses of the other half V_π^{-1} but they are separated by the center braid in which the weft coils around all the cables of equal label. Obviously, a crossing between two particular strings in a braid can be removed if and only if these two strings cross in the opposite manner somewhere else in the braid and these two crossings can be moved adjacent to each other.

Clearly the crossings on either side of the center braid can not be moved to the other side because the coiling of the weft prevents them from being unraveled in this way. Since the cables do not cross each other in the center braid, any weft braid has at least $2inv(V, \pi)c^2$ crossings between the wires for some permutation π.

The center braid consists of crossings between the weft and the wires and the weft crosses nothing outside of the center. Thus the crossings in the center braid may only be removed if they can be canceled within the center braid.

By construction of the center braid in s levels, it is clear that no crossings may be removed between levels. It is also clear that no crossings may be removed within a particular coil. Thus the only way to reduce the number of crossings of the center braid is to change the order in which the coils in each level are made. Since we have s levels of t coils for each collection of $\#(V,i)$ i-cables of c wires each, this takes at least $2tmcs$ crossings (since there are a total of m cables) and the lemma is proved. □

3.3.3 Minimal Weft Braids

We ask under which conditions a weft braid $U \in W_n(r,t,s,c)$ is minimal in length over its equivalence class in $W_n(r,t,s,c)$. The theorem which gives the condition was proved by Paterson and Razborov but we shall give a different proof here.

Theorem 3.3.4 *(Paterson and Razborov [125]) A weft braid $U \in W_n(r,t,s,c)$ in weft form satisfies $L(U) \leq L(U')$ for all $U' \in W_n(r,t,s,c)$ such that $U' \approx U$ if and only if there exists no permutation $\pi \neq \iota$ such that $inv(V,\pi) < inv(V,\iota)$.*

Proof. *(only if)* Suppose there exists a permutation $\pi \neq \iota$ such that $inv(V,\pi) < inv(V,\iota)$. Choose the number of wires per cable $c = rms$. Because of lemma 3.3.1, we have $U_\pi \approx U_\iota$ and due to lemma 3.3.2 we have

$$L(U_\pi) \leq 2inv(V,\pi)c^2 + (r+1+2t)mcs < 2inv(V,\iota)c^2 + 2tmcs = L(U_\iota) \tag{3.4}$$

and U_ι is not of minimal length.

(if) Suppose that U is minimal and that $L(U) < L(U_\iota)$. Then by lemma 3.3.3, we have

$$2inv(V,\pi)c^2 + 2tmcs \leq L(U) < L(U_\iota) = 2inv(V,\iota)c^2 + 2tmcs \tag{3.5}$$

and so $inv(V,\pi) < inv(V,\iota)$ for some permutation π. □

Now we show that NMB ∈ NPC using theorem 3.3.4.

Theorem 3.3.5 *NMB is in NPC.*

Proof. NMB is in NP because the word problem in B_n may be solved in polynomial-time [22]. That is, given a braid, it may be checked in polynomial-time whether it is equivalent to the input braid and if it is of shorter length.

An instance of NMB is specified by a braid group B_n and a word $A \in B_n$. Suppose now that $A \in W_n(r,t,s,c)$. A subproblem of NMB asks whether there exists a weft braid $A' \in W_n(r,t,s,c)$ such that $A' \approx A$ and $L(A') < L(A)$. Theorem 3.3.4 establishes that this is true if and only if there exists a permutation $\pi \neq \iota$ such that $inv(V,\pi) < inv(V,\iota)$ where V is the labeling which defines A in weft form. The question, given $V \in N_r^*$, whether such a permutation exists is the known NPC problem SNMP. Thus NMB contains SNMP as a subproblem. Since NMP ∈ NP and a subproblem of NMB is in NPC, we have that NMB ∈ NPC. □

Even though NMB is an NPC problem, we shall look for an algorithm for it.

3.4 Minimal Length Words

Denote by A_m any braid which satisfies $A_m \approx A$ and $L(A_m) \leq L(A^*)$ for all braids $A^* \approx A$. We now prove a basic lemma which connects A_{max} and A_m. Recall that $A_{max} = \Delta_n^{-s(A)} A'$ where $s(A)$ is the number of inverse generators in A and A' is positive.

3.4.1 Minimal Braids Without Increasing Length

Theorem 3.4.1 *For any braid A, it is possible to obtain A_m from A_{max} by operations which monotonically decrease or keep constant the length of the braid.*

Proof. By construction $A_m \approx A_{max} \approx A$ and A_{max} and A are at least as long A_m. Exponent sum is an equivalence class invariant so that $s(A_m) \leq s(A)$. Replace each inverse generator in A_m with the braid given in proposition 1.3.1 and then use equation (1.12) to bring all the fundamental braids to the front to obtain the braid

$$\begin{aligned} A_{m_{max}} &= \Delta_n^{-s(A_m)} A'_m && (3.6)\\ &\approx \Delta_n^{-s(A_m)} \Delta_n^{s(A_m)-s(A)} \Delta_n^{s(A)-s(A_m)} A'_m && (3.7)\\ &\approx \Delta_n^{-s(A)} \Delta_n^{s(A)-s(A_m)} A'_m && (3.8) \end{aligned}$$

But $A_{max} = \Delta_n^{-s(A)} A'$ and since the braid groups are left-cancelative [64], we have that

$$\Delta_n^{s(A)-s(A_m)} A'_m \approx A' \tag{3.9}$$

with both words positive. Since positive words are positively equal [64], there exists a sequence of braids B_i for $0 \leq i \leq q$ with $B_0 = A'$, $B_q = \Delta_n^{s(A)-s(A_m)} A'_m$, B_j and B_{j+1} different by a single application of the braid group's defining relations and B_i positive for all i. Since exponent sum is an equivalence class invariant, $L(B_i) = L(A')$ for all i.

From A_{max} we may thus reach the form of $A_{m_{max}}$ in equation (3.8) keeping the length of the braid constant. From this form, we may reach A_m by operations which monotonically decrease or keep constant the length of the braid. Thus there exists a sequence of braids W_i for $1 \leq i \leq p$ with $W_0 = A_{max}$, $W_p = A_m$, W_j and W_{j+1} different by a single application of the braid group's defining relations and $L(W_{j+1}) \leq L(W_j)$, which proves the lemma. □

Lemma 3.4.1 basically establishes that we may reach a minimum length representative from A_{max} by rearranging and canceling generators only; it thus, in principle, removes the difficulty we pointed out in the introduction of occasionally having to increase the length before being able to decrease it to an absolute minimum.

3.4.2 The Diagram of a Braid

Garside introduced the notion of a diagram of a *positive* braid in his seminal paper on the braid groups [64]. We present an extension to his construction which draws the diagram of any braid word. The diagram is a list of all those braid words which may be obtained from the given word by rearranging only.

Algorithm 3.4.2 *Input: A braid word A. Output: A list $D(A)$ of all braid words B which may be obtained from A by rearranging of generators only.*

1. Define the diagram of zeroth order as the set $D_0(A) = \{A\}$.
2. The set $D_i(A)$ is obtained from the set $D_{i-1}(A)$ by the following procedure:
   (a) Fix attention on a particular member α of $D_{i-1}(A)$. We read α from left to right and decide at each position whether we may apply any of the moves in equations (3.10) to (3.13).

$$\begin{aligned} \sigma_i\sigma_j &\leftrightarrow \sigma_j\sigma_i \text{ for } |i-j|>1 && (3.10)\\ \sigma_i\sigma_{i+1}\sigma_i &\leftrightarrow \sigma_{i+1}\sigma_i\sigma_{i+1} && (3.11)\\ \sigma_i\sigma_i^{-1} &\leftrightarrow \sigma_i^{-1}\sigma_i && (3.12)\\ \sigma_i\sigma_i^{-1}\sigma_j &\leftrightarrow \sigma_j\sigma_i\sigma_i^{-1} && (3.13) \end{aligned}$$

   (b) If we may, we apply it and store the resultant braid word β in $D_i(A)$ if and only if β is not already contained in $D_j(A)$ for $0 \leq j \leq i$.

(c) We continue to read across α until we have considered all braid words which may be reached from α by a single application of the moves in equations (3.10) to (3.13).

(d) Apply steps (a) through (c) for every braid in $D_{i-1}(A)$. If $D_i(A) = \emptyset$, then the algorithm is done.

3. The diagram $D(A)$ of A is the union of all the $D_i(A)$,

$$D(A) = D_0(A) \bigcup D_1(A) \bigcup \cdots \bigcup D_m(A) \tag{3.14}$$

We show the correctness and termination of this algorithm.

Lemma 3.4.3 *Algorithm 3.4.2 terminates for every A and succeeds in listing all braid words B which may be obtained from A by rearranging of generators only, that is using the braid group relations without introducing or removing any generators.*

Proof. $D_0(A)$ is, by definition, finite. It is obvious that for any braid word of finite length, the moves in equations (3.10) to (3.13) may be applied a finite number of times. Thus, by induction, every $D_i(A)$ is finite. The number of distinct braid words of a given finite length is finite and since the $D_i(A)$ are, by construction, non-overlapping, their union must be finite. Thus there exists an m such that $D_{m+k}(A) = \emptyset$ for every $k > 0$. Thus the algorithm terminates for every A.

The moves listed in equations (3.10) to (3.13) exhaust all possibilities allowed in the braid group under the stipulation that no generators must be removed from or introduced into the word. Thus each word which may be reached from A by rearrangement of generators will eventually be reached by algorithm 3.4.2 and so the algorithm succeeds in listing all the required braid words. □

Lemma 3.4.1 gives the following corollary.

Corollary 3.4.4 *$D(A_{max})$ contains a braid of the form EA_m for $E \approx e$, the identity in B_n.*

Proof. By construction $D(A_{max})$ contains all braid words equivalent to A_{max} by rearranging only. By lemma 3.4.1, A_m can be obtained by a sequence of operations which keeps the length constant or decreases it. Each operation which decreases the length does so by eliminating a subword like $e_i = \sigma_i \sigma_i^{-1} \approx \sigma_i^{-1} \sigma_i$.

Since for all i $e_i \approx e$, the identity in B_n, we have

$$e_i \sigma_j^{\pm 1} \approx \sigma_j^{\pm 1} e_i, \qquad e_i e_j \approx e_j e_i \tag{3.15}$$

for any i and j.

Let us now agree to construct the aforementioned sequence of words without eliminating the subwords e_i but using equation (3.15) to bring them all to the left of the word. At the end, we will obtain a word of the form $A^* = EA_m$ where $E \approx e$ is a braid consisting of all these subwords e_i. The most general form of E is

$$E = e_1^{q_1} e_2^{q_2} \cdots e_{n-1}^{q_{n-1}} \tag{3.16}$$

with $q_i \geq 0$ for all i. So if we could extract E from A_{max}, we would, in the process, obtain A_m. Since the form EA_m is obtained by rearrangements only, $L(E) \leq L(A^*) = L(A_{max})$. This indicates that $\sum_{i=1}^{n-1} 2q_i \leq L(A_{max})$. □

Given a braid A, we thus find A_m by constructing the diagram $D(A_{max})$ and selecting the word with the largest number of cancelation pairs such as $\sigma_i \sigma_i^{-1}$. Clearly there will be more than one braid word for the same number of cancelation pairs. We may agree to choose the least braid word lexicographically for definiteness. It is obvious from the construction that this will be a *unique* form of minimal length for the braid A. We thus have an algorithm to find A_m for any A, it is regrettable that the diagram $D(A_{max})$ is, by construction, very large. Two questions are left to ask: Can we make the result stronger and how large is a typical diagram? These questions will be tackled in the next two sections.

3.4.3 Counterexamples

In theorem 3.4.1 we achieved an upper bound for the necessary increase in length of a braid before it may be reduced to a minimum length. One would like to simplify the result somewhat but we shall show in this section that the two straightforward attempts to simplify or strengthen theorem 3.4.1 are doomed to failure. First we show that we may not, in general, shorten A_{max} to the Garside normal form.

Lemma 3.4.5 *It is not, in general, possible to obtain A_m from $G(A)$, the Garside normal form of A, by operations which monotonically decrease or keep constant the length of the braid.*

Proof. Consider the braid $\alpha = \Delta_3^{-2}\Delta_3\sigma_1^3$. The Garside normal form of α is $G(\alpha) = \Delta_3^{-1}\sigma_1^3$ and the shortest braid which can be obtained from $G(\alpha)$ by rearranging and canceling only is $\alpha' = \sigma_1^{-1}\sigma_2^{-1}\sigma_1^2$. The original form of α is the same as α_{max} and we make the following moves on it

$$\begin{aligned}
\alpha &= \Delta_3^{-2}\Delta_3\sigma_1^3 && (3.17)\\
&\approx \sigma_2^{-1}\sigma_1^{-1}\sigma_2^{-1}\sigma_1^{-1}\sigma_2^{-1}\sigma_2^{-1}\sigma_2\sigma_2\Delta_3\sigma_1 && (3.18)\\
&\approx \sigma_2^{-1}\Delta_3^{-1}\sigma_2^{-2}\sigma_2^2\Delta_3\sigma_1 && (3.19)\\
&\approx \sigma_2^{-1}\sigma_1 && (3.20)
\end{aligned}$$

which is shorter than α' and is in fact the minimal length of this 3-braid. This provides an example for which the minimal length is not obtainable from the Garside normal form of the braid by rearranging and canceling only and thus proves the lemma. □

One may think that it would be sufficient to list the diagram of the negative and positive subbraids of A_{max} and search for a maximal length subbraid which is common to the end of the first and the beginning of the second diagram but this is not true as the following lemma shows.

Lemma 3.4.6 *There does not exist an A_m in the form A_1A_2 with A_1 negative and A_2 positive for every A.*

Proof. Consider the braid $A = \sigma_1^{-1}\sigma_2\sigma_1^{-1}$, the Garside normal form of which is $G(A) = \Delta_3^{-2}\sigma_2\sigma_1\sigma_1\sigma_1\sigma_2$. In fact, A is already minimal as can be seen by Berger's algorithm [17] or by using the above procedures. We wish to show that there does not exist another braid equivalent to A of length three in the form A_1A_2 with A_1 negative and A_2 positive. Since exponent sum is a conjugacy class invariant, we need only check eight cases. Below we list the eight 3-braids of length three and exponent sum -1 in the required form and their Garside normal forms.

$$\begin{aligned}
\sigma_1^{-1}\sigma_1^{-1}\sigma_1 &\rightarrow \Delta_3^{-1}\sigma_1\sigma_2 && (3.21)\\
\sigma_1^{-1}\sigma_1^{-1}\sigma_2 &\rightarrow \Delta_3^{-2}\sigma_2\sigma_1\sigma_1\sigma_2\sigma_2 && (3.22)\\
\sigma_1^{-1}\sigma_2^{-1}\sigma_1 &\rightarrow \Delta_3^{-1}\sigma_1\sigma_1 && (3.23)\\
\sigma_1^{-1}\sigma_2^{-1}\sigma_2 &\rightarrow \Delta_3^{-1}\sigma_1\sigma_2 && (3.24)\\
\sigma_2^{-1}\sigma_1^{-1}\sigma_1 &\rightarrow \Delta_3^{-1}\sigma_2\sigma_1 && (3.25)\\
\sigma_2^{-1}\sigma_1^{-1}\sigma_2 &\rightarrow \Delta_3^{-1}\sigma_2\sigma_2 && (3.26)\\
\sigma_2^{-1}\sigma_2^{-1}\sigma_1 &\rightarrow \Delta_3^{-2}\sigma_1\sigma_2\sigma_2\sigma_1\sigma_1 && (3.27)\\
\sigma_2^{-1}\sigma_2^{-1}\sigma_2 &\rightarrow \Delta_3^{-1}\sigma_2\sigma_1 && (3.28)
\end{aligned}$$

Since the Garside normal form solves the word problem and none of the above Garside normal forms are identical to $G(A)$, the braid A does not possess a minimal length representative in the required form. □

Table 3.1: The Size of Diagrams of Fundamental Words

n	p	$\lvert D\left(\Delta_n^p\right)\rvert$	max. i
3	1	2	1
3	2	8	2
3	3	38	5
3	4	196	8
3	5	1062	13
3	6	5948	18
3	7	34120	25
4	1	16	7
4	2	1654	15
5	1	768	25

3.5 The Size of Diagrams

Let a be a n-braid of length L with diagram $D(a)$. Consider the braid $a' = a\sigma_i\sigma_i^{-1}$ for some $1 \leq i < n$. We are concerned with the size of $D(a')$ in terms of the size of $D(a)$. For each member of $D(a)$, the cancelation pair $\sigma_i\sigma_i^{-1}$ may appear in any place in both possible orders ($\sigma_i\sigma_i^{-1}$ and $\sigma_i^{-1}\sigma_i$), so in $2(L+1)$ positions. There may be further moves possible by use of the braid group relations but the number of these are clearly bounded by a function linear in L. So the diagram of a word will increase in size by a factor linear in its length for each possible cancelation pair. Given a random positive n-braid a of length L, how many members will $D(a)$ have, on average? We conjecture that

Conjecture 3.5.1 *For any braid $a \in B_n$ of length L, we have that $|D(a)| \leq |D(\Delta_n^p)|$ with $p = \lceil 2L/(n(n-1))\rceil$.*

Conjecture 3.5.1 would provide an upper bound for the size of the diagram of any word in terms of the diagrams of the diagrams of Δ_n^p which topogically are a series of p half-twists of the braid strings about the vertical axis. In extensive computer simulations, the conjecture was checked and seems to hold. What it seems to indicate is that the half-twist has the most topological freedom for its length and number of strings under the constraint that the crossing number must be kept constant. This is quite intuitive, yet the conjecture seems to be difficult to prove.

We have investigated the diagrams of several Δ_n^p for their size and for the distribution of braids over the subdiagrams at each stage of the construction in algorithm 3.4.2. In table 3.1 we list the size of the diagram and maximal subdiagram index for p half-twists on n strings.

We conclude that the diagram of a typical braid word grows exponentially with its length and braid index and thus our method of finding the minimal length braid word equivalent to a given braid has exponential complexity. This is not surprising as we showed that the problem is NP-Complete. We shall give a heuristic algorithm and other methods in the next chapter. The properties of the braid groups that made the above solution possible are: (i) It is possible to write all inverse generators as products of the generator of the center and a positive word, (ii) the defining relations relate positive words only and (iii) the braid groups are right and left-cancellative. It is likely that any group which has these properties, has an analog of the Garside normal form and has a solution to the minimum word problem similar to the one above.

Chapter 4

Minimal Words via Elastic Relaxation

In this chapter, we investigate several different approaches to obtain minimal configurations: we employ three different relaxation techniques and compare these with each other and with an algebraic heuristic algorithm, in terms of minimization (of energy and crossing number) and time efficiency. By energy we mean total string length of the braid. It is found that more than half of the crossings of a sufficiently large braid (in terms of crossing number and number of strings) are redundant. We analyze the different methods and say in what circumstances which method is to be favored and conclude that minimum braid energy and minimum braid crossing number are substantially different measures of topological complexity for braids.

The main purpose of this chapter is to propose an efficient tool for finding minimal braid configurations. With this in mind, we first demonstrate how to generate random braids both algebraically and geometrically and how to convert between them in §2. In §3 and §4, we introduce the forces which will minimize the crossing number and the elastic energy respectively. An algebraic heuristic minimization is given in §5 and the results of computational experiments of all approaches are discussed in §6 together with theoretical comparisons. We conclude in §7.

4.1 Introduction

Topological constraints appear in many complex systems. In biology the amount of twisting and knotting of DNA molecules can affect molecular interactions and dynamics [90]. In polymer physics the degree of entanglement of the polymer filaments helps to determine the elastic properties of the polymer [6]. In astrophysics, applications involve the behavior of magnetic fields (such as those found in stars and accretion disks) with complex topologies [123, 16, 17, 36, 124]. In dynamical systems theory, the time history of the system can be represented by a set of braided particle orbits; the topology of the braid reflects qualitative aspects of the dynamics [46, 34, 108, 113]. In turbulence theory the degree of entanglement of the vortex lines provides a statistical measure of flow properties; this measure is distinguished from most others used in turbulence by being based on the flow in real space rather than on the spectral transform of the flow in Fourier space. In statistical mechanics, braid and knot theory has significantly contributed to exactly solvable models via knot polynomials, for example [152]. Random knotting, as opposed to the random braiding discussed here, has also been investigated [134].

All these applications involve a set of curves (e.g. long molecules, magnetic or vortex lines) which are knotted, linked, or braided. Knot theorists have devoted great effort to classifying such objects. One important part of this effort concerns finding measures of complexity. This idea goes back at least as far as Tait, who first set out tables of different knot types [143]. Tait organized the knot tables according to a simple complexity measure, the minimum crossing number C_{min}. This number gives the minimum number of crossings of a knot as seen in any two dimensional projection.

There are two types of topological invariants. The first, sometimes called *isotopic invariants*, involve quantities that remain constant if we deform the set of curves. Examples include the Gauss

linking integral, helicity integrals, and knot polynomials. The second type involves quantities which do change when the curves deform, but have a lower bound. This second type of invariant can be regarded as a measure of topological complexity [136] of which both crossing number and energy are examples.

This chapter investigates the crossing number and energy for random braids. Braids consist of a set of curves stretching between two parallel planes. The endpoints of the curves are fixed but, between the two planes, the curves are free to move so long as they do not cross through each other. Braids are important in knot theory because (unlike knots) they can be readily classified using group theory [22]. They are also important in solar physics, as the field lines within a coronal magnetic loop are braided. (In fact, a coronal loop forms an arch with both ends in the photosphere. But a simple geometrical transformation straightens out the arch into a cylinder with ends on two parallel planes.)

Energy, of course, has the most immediate physical significance. For example, a solar magnetic loop usually stays close to the energy minimum (equilibrium) state consistent with the field topology. Sometimes this equilibrium becomes unstable; a rapid reconnection event changes the topology and energy is released in a flare. Crossing number, on the other hand, relates more directly to the geometry of the field lines. The state of minimum crossing number may not be exactly the minimum energy state, but one may conjecture that they will be close. We can, in fact, find strict lower bounds for the energy of a magnetic field given its crossing number. This has been done for fields in a spherical geometry with closed field lines [62] and in a cylindrical geometry with braided field lines [16].

We can also consider continuous fields rather than knotted or linked curves, e.g. knotted fluid flows and magnetic configurations [111, 112, 62, 44, 16]. Crossing number can be defined for a continuous field by averaging the crossing number of all pairs of field lines [62, 16]. This average crossing number will have some positive minimum amongst all fields with the same field line topology. Minimum crossing number and minimum field energy will then both measure the topological complexity of the field. For example, a closed, knotted tube of magnetic flux will have a magnetic energy which generally increases with the C_{min} of the knot (and with internal twist of field lines inside the tube).

There has recently been a major effort to find the ideal shapes of knots. While the definition of "ideal" varies, the ideal shape is mostly obtained by minimizing some form of knot energy. Various energy functionals have been suggested for knotted curves [62, 90, 137]. These energy functionals have a positive minimum depending on the knot type analogous to minimum crossing number. Some theoretical questions arise from this work. For example, are the energy minima found using these approaches local or global minima? One would like an energy such that the minimum is global for any initial configuration. This does not seem to be possible, however [52]. Another important question is whether an energy minimum corresponds to the minimum of a more traditional measure of complexity, for example the crossing number. As mentioned above, it has been implicitly assumed in the literature that this is true, however we argue here on the basis of statistical results that this is not true. As mentioned above, energy can be defined in many ways and different energies behave differently. We consider energy to be the length of the strings in the braid.

4.2 Obtaining and Embedding Random Braids

For the results of our comparison of minimization strategies to be useful we discuss our approach of generating random braids in a way such that we obtain a representative selection of braids. First, we discuss how to obtain a random algebraic braid, then how to convert it into a geometrical braid and finally how to reverse this conversion.

4.2.1 Randomly Generating Algebraic Braids

An algebraic n-braid is a word over the generators of the braid group B_n, that is the set $\{\sigma_i^{\pm 1}\}$ for $1 \leq i < n$. Given a number of strings n and a number of crossings c, there are clearly $(2(n-1))^c$ algebraic braids possible. Not all of these braids are necessarily topologically distinct however. Two words in B_n represent the same braid if and only if one can be transformed into the other

using the following relations

$$\sigma_i \sigma_i^{-1} = e, \tag{4.1}$$

$$\sigma_i \sigma_j = \sigma_j \sigma_i \qquad |i-j| > 1, \tag{4.2}$$

$$\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1} \tag{4.3}$$

where e is the identity in B_n (topologically e is the braid of no crossings; it consists of n vertical strings). The problem of determining whether two words are equivalent or not is known as the word problem, the most efficient algorithm for which was found by Birman, Ko and Lee [22] and runs with complexity $O(nc^2)$.

We may select a word at random from the $(2(n-1))^c$ possibilities by choosing each of the c generators at random from the set $\{\sigma_i^{\pm 1}\}$ for $1 \leq i < n$. The number of words in the set of the $(2(n-1))^c$ possibilities corresponding to a particular braid will not, on average, depend on this braid; that is, the intersections of the equivalence classes of the represented braids and our set are, on average, of equal size. Thus the uniformly random word corresponds to a uniformly random braid.

4.2.2 Embedding Algebraic Braids

We describe how to determine the geometric braid which an algebraic braid represents in this section. If we select a random braid in the above algebraic manner, we must embed it in three-dimensional Euclidean space in order to use an energy minimizing algorithm on it.

An n-braid consists of n strings embedded in R^3 and thus we may describe the braid by giving n vector functions $\vec{x}_i(z,t)$ parametrized by the vertical z coordinate and time t. We construct these functions such that

$$\vec{x}_i(z,t) = (x_i(z,t), y_i(z,t), z)\,. \tag{4.4}$$

The vertical component of the functions is assumed independent of time. This is needed only for efficient computer implementation of the model since it ensures that the points by which the functions would have to be approximated do not collect near the ends of the strings during the simulation, as they would without this constraint. We shall take the braid to lie between $z = 0$ and $z = 1$ and, in keeping with the definition of braid isotopy, the points $\vec{x}_i(0,t) = \vec{x}_i(0)$ and $\vec{x}_i(1,t) = \vec{x}_i(1)$ will be independent of time.

We will generate b points per crossing and string in order to represent the braid; thus there will be bc points per string in the braid. The position vector of the j^{th} point on string i is $\vec{x}_i(j/bc, t)$, which must be specified at time $t = 0$. If we are given a braid word in the generators σ_i, we set

$$\vec{x}_i(0,0) = (\delta_x(i-1), 0, 0) \tag{4.5}$$

where δ_x is a given parameter. The subsequent points will have a z coordinate between 0 and 1. We then read the first generator in the braid word and add b points to all strings not involved in the crossing which are vertically above the last set point. For the strings involved in the crossing, the x and z coordinates are simply straight lines exchanging the two strings over a vertical region of size $1/c$. The y coordinate is constructed from the Gaussian distribution

$$y_i(j/bc, t) = \pm \frac{\delta_x}{3} \exp\left(-120 \frac{[\alpha/2 - k\alpha/(b-1)]^2}{\alpha^2} \right) \tag{4.6}$$

where $\alpha^2 = \delta_x^2 + 1/c^2$ and k runs from zero to b as j increases. The positive y coordinate is chosen for the overcrossing string and the negative y coordinate for the undercrossing string. See figure 4.2 for two examples.

It has been found in practice that this yields an esthetically pleasing embedding of the braid. Physically, the Gaussian distribution is also intuitive because we assume the string to be elastic later. Numerically, it may be seen that the initial and final points of the Gaussian are always close enough to the straight segments of the braid that this does give a good distribution of points over the braid.

4.2.3 Extracting Geometric Braids

In this section we describe how to reverse the process described in the last section. In our model, each point will move on a horizontal plane, the original vertical spacing between beads is preserved. Thus we can slice the final braid into $bc - 1$ slices, which are the segments of the braid strings between two points. Given a position of the observer, it is then easy to extract the left-to-right order of the braid strings for each slice. At the start we label each string with a number increasing from 1 for the leftmost string to n for the rightmost string. The initial order is thus the identity permutation on n elements and each slice will have an associated permutation.

In this way we build up a list of permutations from the start to the end of the braid. If the permutation is the same as the one before it, no further generators need to be inserted into the braid word. If it is different, then we must insert a generator into the braid word. Suppose that strings i and j are switched in one transition between permutations. We must establish which string is closer to the observer on the current vertical level, which we can readily do. We must also determine which of i and j appears first in the permutation structure from the left and in which position it occurs. Suppose that i is found first in position k in the permutation and that i passes over j, then we must add the generator σ_k to the Artin word. If i passes under j, then we must add σ_k^{-1}.

If there is at most one transposition of elements in the permutation at every step, the translation is simple. If there are more than one, we must be careful to assess which string crossed with which other string. This however may be done simply by checking which string was closest to a given string at the current vertical level. This leaves us with the remote possibility of a triple point, that is three strings crossing at once. This sort of crossing may be removed by a slight shift in the observer's position. However, a repulsive force which we will introduce later and keeps braids from overlapping effectively negates the possibility of triple points and so the only case of complex transitions left is several exchanges, which could be determined from the string positions themselves.

In practice, this recognition algorithm has worked well. For a given embedding of a topological braid, the crossing number depends upon the observer's position, in general. To take this into account, we rotate the observer around the braid and compute the braid word for many observation angles and choose the shortest braid word.

4.3 A Crossing Number Minimizing Force

Here, we will obtain a force which directly minimizes the crossing number of a braid for later comparison with the energy minimizing forces and the algebraic approach.

4.3.1 Expressions for crossing number

Recall that a braid is represented by a set of n curves $(x_i(z,t), y_i(z,t), z)\,, i = 1, \ldots n, 0 \le z \le 1$. By projecting the curves onto a vertical plane we can detect a number of crossings. Let the projection angle be ϕ, with direction vector $\vec{p}(\phi) = (\cos\phi, \sin\phi, 0)$. Thus for $\phi = 0$ the projection plane will be the x-z plane. The crossing number $C(\phi)$ will then be a function of ϕ. If we distort the braid, $C(\phi)$ will change. For fixed ϕ we can use group theory to minimize $C(\phi)$ over all possible deformations of the braid [17]. For $n \le 3$ an algorithm linear in the number of initial crossings is known [17] but no efficient algorithm for $n > 3$ is known (see §5 for more details). Thus, it is worth it to pursue numerical relaxation methods similar to energy relaxation but specifically based on crossing number minimization.

The crossing number dependence on projection angle ϕ can be removed by choosing the minimum all projection angles:

$$C_{min} \equiv \min_{\phi=0}^{\pi} C(\phi). \tag{4.7}$$

The crossing number is a sum over pairs:

$$C \equiv \sum_{ij} C_{ij} \equiv \sum_{i=1}^{n} \sum_{j=i+1}^{n} C_{ij} \tag{4.8}$$

where C_{ij} just counts crossings between strings i and j.

In addition to decomposing C into contributions from each ij pair, we can look at how C increases with z. Thus we will let $C_{ij}(z_0)$ measure the crossings of strings i and j between $z = 0$ and $z = z_0$. In this notation $C_{ij} = C_{ij}(1)$. Let $\vec{x}_i(z) = (x_i(z), y_i(z), z)$ and define relative position vectors and angles

$$\vec{r}_{ij}(z) \equiv \vec{x}_j(z) - \vec{x}_i(z); \tag{4.9}$$

$$\theta_{ij}(z) \equiv \tan^{-1}\left(\frac{y_j(z) - y_i(z)}{x_j(z) - x_i(z)}\right). \tag{4.10}$$

We will let a prime denote differentiation by z, for example

$$\theta'_{ij}(z) \equiv \frac{d\theta_{ij}(z)}{dz} = \frac{\epsilon_{\alpha\beta} r^\alpha r'^\beta}{r^2} \tag{4.11}$$

where $\epsilon_{\alpha\beta}$ is the Levi-Civita tensor and $r'^\beta = dr^\beta/dz$.

If strings i and j wind around each other near some height z then $|\theta'_{ij}(z)| > 0$. Also, there will be some projection angles ϕ where the strings will be seen to cross (in fact, a crossing will be seen at height z if $\vec{p}(\phi) = \vec{r}_{ij}(z)$). It can be shown [16] that

$$\frac{dC_{ij}(z)}{dz} = \frac{1}{\pi}\left|\theta' ij(z)\right|. \tag{4.12}$$

We now have

$$C = \frac{1}{\pi}\sum_{ij}\int_0^1 \left|\theta' ij(z)\right| dz. \tag{4.13}$$

4.3.2 Derivation of the crossing number force

Suppose we employ C as a potential energy term in a Lagrangian for n strings. Varying the Lagrangian will give an equation of motion with a force corresponding to C. Adding a strong damping force will then give us dynamics which can be followed numerically to relax the strings to an 'equilibrium' state, i.e. a state which is at least a local minimum of C. Let μ be mass per unit length and consider a time interval T. The Lagrangian action should then be

$$S = \int_0^T (K - \lambda C) dt \tag{4.14}$$

where

$$K = \frac{1}{2}\int_0^1 \mu\left(\frac{d\vec{x}(z,t)}{dt}\right)^2 dz \tag{4.15}$$

is the kinetic energy term and λ is a constant. The K variation (at height z) gives the usual acceleration term $\mu\, d^2\vec{x}/dt^2$.

However, the variation of C will not work without modification. There are two problems. First, there is the presence of the absolute value in equation 4.13; derivatives will be ill defined at singular points where $|\theta' ij(z)| = 0$. In between these singular points we could replace the absolute value by a factor ± 1. However, if we do so an even nastier problem arises: $\pm d\theta_{ij}(z)/dz$ is a total differential. But the variation of a total differential vanishes apart from boundary terms.

Fortunately, there is a simple way out of these difficulties: replace $|\theta'_{ij}|$ with

$$X_{ij} = \sqrt{(\theta' ij)^2 + \epsilon^2}. \tag{4.16}$$

Later we can let $\epsilon \to 0$. Let $\vec{F}_{ij}$ be the force on string i due to string j associated with the potential λX_{ij}. A variation in $\delta\vec{x}_i$ and $d(\delta\vec{x}_i)/dz$ leads to

$$\vec{F}_{ij} = -\lambda\left(\nabla_i X_{ij} - \frac{d}{dz}\nabla'_i X_{ij}\right); \qquad \nabla'_i \equiv \frac{\partial}{\partial \vec{x}'_i}. \tag{4.17}$$

It simplifies the calculation to note that

$$\theta'_{ij} = \vec{x}'_i \cdot \nabla_i \theta_{ij} + \vec{x}'_j \cdot \nabla_j \theta_{ij}. \tag{4.18}$$

so, evaluated at the point $(\vec{x}_i, z)$

$$\nabla'_i \theta' ij = \nabla_i \theta_{ij}. \tag{4.19}$$

Thus, suppressing the i, j labels,

$$\begin{aligned} \frac{d}{dz}\nabla' X &= \frac{d}{dz}\left(\frac{\theta'\nabla'\theta'}{X}\right) = \frac{d}{dz}\left(\frac{\theta'\nabla\theta}{X}\right) && (4.20)\\ &= \frac{1}{X}\left(\theta''\nabla\theta + \theta'\nabla\theta' - \frac{\theta'^2\theta''\nabla\theta}{X^2}\right) && (4.21)\\ &= \frac{\theta'\nabla\theta'}{X} - \epsilon^2\left(\frac{\theta''\nabla\theta}{X^3}\right). && (4.22) \end{aligned}$$

The first term here cancels the ∇X term in 4.17. 4.17 becomes

$$\vec{F} = -\lambda\epsilon^2\left(\frac{\theta''\nabla\theta}{X^3}\right). \tag{4.23}$$

We let $\lambda = \epsilon^{-2}$ and let $\epsilon \to 0$. Summing over all strings j gives the crossing force on string i,

$$\vec{F}_i = -\sum_{j\neq i}\left(\frac{\theta_{ij}{}''\nabla\theta_{ij}}{|\theta' ij|^3}\right). \tag{4.24}$$

4.3.3 Simulation Considerations

For numerical purposes it may be wise to retain a small ϵ as a softening parameter, i.e. replace $|\theta' ij|^{-3}$ by X_{ij}^{-3}. This will prevent the force blowing up near $\theta' ij = 0$. Being as explicit as possible, the force on the point $\vec{x}_i(z)$ of the braid is

$$\vec{F}_i(z) = -\lambda\epsilon^2 \sum_{j\neq i} \frac{\left[(2r'\vec{r}' - r\vec{r}'')\cdot(\hat{z}\times\vec{r})\right](r\hat{z}\times\vec{r})}{\left((\vec{r}'\cdot\hat{z}\times\vec{r})^2 + r^4\epsilon^2\right)^{3/2}} \tag{4.25}$$

where $\vec{r} = \vec{r}_{ij}(z)$, $r = \sqrt{\vec{r}\cdot\vec{r}}$, $r' = (\vec{r}'\cdot\vec{r})/r$ and $\hat{z}$ is the unit vector in the z direction.

What we observe in practice is that this force causes the strings of the braids to move apart from each other and prevents equilibrium from being reached. Thus we apply the additional constraint that

$$\sum_i \vec{x}_i(z)\cdot\vec{x}_i(z) \leq R \tag{4.26}$$

where R is a parameter of the model. After imposing this we can agree to have reached equilibrium if and only if the maximum distance moved by a point on the braid at any time step is less than another parameter η. We discuss the consequences of the choices for these two parameters in §6.

4.4 Energy relaxation

Two energy minimizing approaches were tested with respect to minimizing crossing number. Both minimize elastic energy but they differ essentially in the way the elastic force is implemented: a nearest neighbor approximation (the *constrained* elastic force) versus a tension force depending on the curvature (the *curvature* elastic force). As these forces treat the strings as elastic, they pull the strings closer together and would cause them to intersect and thereby change topology. In order to prevent this, we shall introduce a repulsive force $\vec{F}_i^{(r)}(z,t)$ to the elastic force $\vec{F}_i^{(e)}(z,t)$ to make up the total force which we use to simulate the braids,

$$\vec{F}_i(z,t) = \vec{F}_i^{(e)}(z,t) + \vec{F}_i^{(r)}(z,t). \tag{4.27}$$

For the purposes of the repulsive force, we imagine the strings to be of circular cross-section with diameter d. We define this repulsive force by

$$\vec{F}_i^{(r)} = \sum_{k \neq i} \begin{cases} 0 & \text{for } |\vec{x}_i - \vec{x}_k| > d \\ \frac{\vec{x}_i - \vec{x}_k}{|\vec{x}_i - \vec{x}_k|} \left(d - |\vec{x}_i - \vec{x}_k|\right) & \text{otherwise.} \end{cases} \tag{4.28}$$

Since the repulsive force is non-zero in only a limited number of cases, computing it is relatively fast as opposed to using a potential function.

4.4.1 The Constrained Elastic Force

If we imagine the points of the geometric braid to be beads of mass m connected by springs of spring constant k and zero natural length, the elastic force on the j^{th} bead due to the two springs attached to it is (considering only nearest neighbor interactions)

$$\mathbf{F}_i^{(e)}\left(\frac{j}{bc}, t\right) = -k\left(2\vec{x}_i\left(\frac{j}{bc}, t\right) - \vec{x}_i\left(\frac{j+1}{bc}, t\right) - \vec{x}_i\left(\frac{j-1}{bc}, t\right)\right). \tag{4.29}$$

This is the constrained elastic force. As given in equation 4.29 the constrained elastic force is a finite difference scheme for the differential equation

$$\mathbf{F}_i^{(e)}(z,t) = \frac{k}{b^2c^2}\frac{d^2\vec{x}_i(z,t)}{dz^2}. \tag{4.30}$$

Once the total force is known, we apply it to the beads

$$\vec{x}_i(z, t+\delta_t) = \vec{x}_i(z,t) - \frac{\mathbf{F}_i(z,t)}{2m}(\delta_t)^2. \tag{4.31}$$

We have neglected the fact that beads should acquire a velocity after the force is first applied. Ignoring this velocity serves to heavily damp the system, which is desirable for the simulation. A proof that this is acceptable on a fundamental level is given in [11] and references therein. The force is applied for a duration of δt after which the beads will have moved a certain distance. The maximum distance moved by any bead in the whole braid during any step $r(t)$ decreases monotonically to zero since the system is heavily damped due to the neglection of the velocity and that fact that the springs have natural length zero. If no bead moves more than a minimum distance of η, we may terminate the simulation because in all subsequent steps of the simulation no bead will move further than η. Thus the end of the simulation is reached when $r(t) \leq \eta$.

A given braid will determine n and c but we have endowed the model with a number of parameters: The string diameter d, the number of beads per crossing b, the mass of a bead m, the spring constant k, the separation of the strings δ_x, the duration of the force δ_t and the equilibrium distance η. Based on computer experiments we make choices for some of these

$$d = \frac{\delta_x}{6}, \qquad \delta_t^2 = 2m, \qquad \eta = \frac{\delta_x}{10^5}, \tag{4.32}$$

$$0.1 \leq k < 0.5, \qquad 10 \leq b \leq 50, \qquad \delta_x = \frac{1}{n-1}. \tag{4.33}$$

These values have been found to give good results. With increasing k, fewer steps are required to reach equilibrium but $k < 0.5$ must be observed because otherwise the repulsive force will not be very successful. Accuracy increases with b but so does the computation time. Setting $b < 10$ will fail because the distances between beads are large enough for the repulsive force not to guarantee isotopy, however $b > 50$ is unnecessarily expensive in terms of time.

4.4.2 The Curvature Elastic Energy

The other way of dealing with elastic relaxation is to treat each string in the braid as a bungee cord, subject to a tension force which aims to reduce any curvature and bring back the string to a straight configuration (given the constraint on the end points). Indeed, as already remarked above,

a repulsive force among the strings is needed to counteract the tension in order to maintain the topology unchanged.

An expression for the elastic force can be readily obtained by using a variational approach similarly to what has been done in §3. In fact, since this force tends to minimize the length of the bungee cord, we can employ the total length of the strings as a potential energy term in the Lagrangian action S describing the braid (4.14)

$$S = \int_0^T (K - \lambda L) dt \tag{4.34}$$

where

$$K = \frac{1}{2} \int_0^1 \mu \left(\frac{d\vec{x}(z,t)}{dt} \right)^2 dz \tag{4.35}$$

is the kinetic energy term, λ is a constant,

$$L = \int ds \ , \tag{4.36}$$

and ds is the infinitesimal arclength along each string ($ds^2 = dx^2 + dy^2 + dz^2$). The variation of the action (4.34) provides an equation of motion which is the well-known equation for the vibrating cord. Any perturbation from the equilibrium position is opposed by a restoring force proportional to

$$\mathbf{F} \ \propto \ \frac{d^2\vec{x}}{ds^2} \tag{4.37}$$

and always directed along the radius of curvature. However, we are not allowed to perform any displacement along the z direction because the braid is represented by a set of curves $(x_i(z,t), y_i(z,t), z)$, $i = 1, \ldots n$, $0 \le z \le 1$. Thus, instead of (4.37), we use the *horizontal* force

$$\mathbf{F}_i^{(e)} = \frac{d^2\mathbf{x}_i}{ds^2} - \frac{d^2 z}{ds^2} \frac{d\mathbf{x}_i}{dz} \ . \tag{4.38}$$

This is the curvature elastic force. This force moves the curve as the full curvature force would; the second term gives an extra horizontal displacement to the string which compensates for the effect of the missing vertical force. Once its value is known in each point of the braid and cumulated with the repulsive term, advancing in time is achieved according to the same scheme as above (4.31).

The actual evaluation of the curvature elastic force involves the computation of second and first order spatial derivatives. In this case, then, we found it convenient to use a grid of N evenly spaced points along the z axis and ordinary centered difference. Stopping criteria for the numerical simulation of energy relaxation were defined as explained in the previous section. We kept $N = 200$ in all the cases presented in the following, while the choice of the other parameters was

$$d = \delta_x, \qquad \delta_t^2 = 2m, \tag{4.39}$$

$$\eta = \frac{\delta_x}{10^5}, \qquad \delta_x = \tfrac{1}{20} \ . \tag{4.40}$$

4.5 Algebraic Minimization

Recall that the braid group B_n is defined by

$$B_n \ = \langle \quad \{\sigma_i\} : \ 1 \le i < n; \tag{4.41}$$

$$\sigma_i \sigma_j = \sigma_j \sigma_i \ |i-j| > 1; \sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1} \rangle \ . \tag{4.42}$$

An n-braid A of c crossings is a word in B_n of word-length c, so the general form of A is

$$A = \sigma_{a_1}^{\epsilon_1} \sigma_{a_2}^{\epsilon_2} \cdots \sigma_{a_c}^{\epsilon_c} \qquad \epsilon_k = \pm 1, \ 1 \le a_k < n, \ \forall k : 1 \le k \le c. \tag{4.43}$$

Consider an n-braid A of the form given in equation 4.43. Suppose we wish to find the n-braid A_m equivalent to A such that the length $L(A_m)$ of A_m is minimal over the equivalence class of A. It has been shown [125] that this question is NP-complete and hence computationally difficult (if P $\neq$ NP, it is intractable). The following presents a heuristic algorithm for getting close to A_m. We begin with the leftmost generator of A and attempt to move it to the right using both braid group operations. If we can cancel it along the way, we do and if we can not, we move it back to where it started. In this way, we proceed to move all the generators as far to the right as possible. Then we begin at the end and move each generator as far to the left as possible in the same manner. This algorithm will always produce an equivalent braid A' such that $L(A') \leq L(A)$. We consider $L(A)$ generators and move them $O(L(A))$ moves to the right and left. Thus this algorithm takes $O\left(L(A)^2\right)$ time and constant memory. In fact we move a particular generator at most $L(A)$ generators and this is only for the case when all the other generators commute with it, thus the average case complexity is likely to be close to linear in $L(A)$.

This algorithm will not produce a minimum length representative in all cases because it can not unravel complex crossings. To get to the minimum length would require more subtle transformations than just movements to the right or left, which topologically correspond to pulling the strings apart from underneath the crossing. However, as computer experiments show, it does do quite well.

Let us calculate an upper bound to the reduction ratio obtained by this method as a function of n and c. To calculate these, consider the likelihood that a particular generator will be followed by its inverse, which is just $Q_0 = 1/2(n-1)$. The probability Q_j that a generator and its inverse are separated by j generators through which either can be moved is the corresponding probability for $j = 1$ to the power j. We require the number of braids of length 1 which may be generated so as not to contain the generator interfering with the movement of generator σ_i. If $i = 1$ or $n-1$, this is $2(n-3)$ and $2(n-4)$ otherwise. Thus

$$Q_j = \left[\frac{2(n-4)\left(\frac{2(n-1)-2}{2(n-1)}\right) + 2(n-3)\left(\frac{2}{2(n-1)}\right)}{2(n-1)}\right]^j Q_0 \tag{4.44}$$

$$= \left[\frac{n^2-5n+5}{(n-1)^2}\right]^j Q_0 \tag{4.45}$$

The final factor of Q_0 is present because the generator after the sequence of j generators is required to be inverse of the original generator, an event with probability Q_0. To get the total probability Q of being able to cancel a generator σ_i with its inverse by simple exchange movements over the length $j = 0, 1, \cdots$, we must sum these probabilities in order weighted by the probability that their predecessors did not happen. Thus

$$Q = Q_0 + (1-Q_0)Q_1 + \cdots + \prod_{k=0}^{j-1}(1-Q_k)Q_j + \ldots \tag{4.46}$$

Note that since the exchange move is not allowed for $n = 3$, $Q = Q_0$ for $n = 3$. The reduction ratio R which occurs as a consequence of this probability is $R = 1 - 2Q$ since each time that the event happens two generators may be canceled. Note that in this calculation we have considered the probability that a generator can be moved next to its inverse in the word using only the far commutation relation that $\sigma_i\sigma_j = \sigma_j\sigma_i$ for $|i-j| > 1$ in a long braid. The heuristic algorithm however uses both braid group moves to attempt to move generators next to their inverses. Thus R is an upper bound for the reduction ratio achieved by the heuristic algorithm as the braid becomes long.

In §6 we present the results of the algebraic reduction of a large number of braids but a few comments about the efficiency of the algorithm are in order. The only exact algorithm to minimize braid is valid only for $n \leq 3$ [17] and by comparing this heuristic to this exact algorithm, we find that the heuristic finds a braid the length of which is within five percent of the length found by the exact algorithm and that it reaches the actual minimum in 0.005 of all cases. This shows that the heuristic is quite effective for $n = 3$ (note that reduction for $n = 1, 2$ is trivial since B_1, B_2 are free groups).

Table 4.1: Reduction ratios $\alpha = C_{min}/C(0)$ as a function of the number of strings n.

n	Heuristic	Heuristic Bound	Constrained Elastic	Curvature Elastic	Crossing Force
3	0.328	0.500	0.412 ± 0.028	0.516 ± 0.007	0.426 ± 0.006
4	0.525	0.632	0.483 ± 0.021	0.611 ± 0.006	0.430 ± 0.005
5	0.579	0.652	0.529 ± 0.016	0.658 ± 0.005	0.447 ± 0.004
6	0.609	0.662	0.555 ± 0.012	0.676 ± 0.005	0.455 ± 0.004
7	0.627	0.668	0.574 ± 0.010	0.692 ± 0.005	0.469 ± 0.004
8	0.640	0.673	0.590 ± 0.008	0.694 ± 0.005	0.471 ± 0.004
9	0.650	0.677	0.600 ± 0.007	0.697 ± 0.005	0.476 ± 0.004
10	0.658	0.681	0.617 ± 0.006	0.693 ± 0.004	0.481 ± 0.004

4.6 Some numerical results

Extensive numerical calculations were made in order to compare the above methods of finding the minimum crossing number of a braid. A large selection of random braids were generated as discussed in §2 and then simulated using all four different methods described above. In §6.1, the comparison is made in terms of the ratio α of final crossing number to initial crossing number as a function of number of strings and number of initial crossings. When braids were reduced using forces, we chose the final braid by using the translation algorithm described in §2.3 for many angles around the vertical axis of the braid and isolated the braid with minimum crossing number. The reason for this is that the position of the observer affects the crossing number seen from that perspective. In §6.2, we compare the efficiency of our methods, giving an estimate of the algorithmic complexity in terms of the number of strings n and the other free parameters. Finally in §6.3, we focus on the effect of the three forces on the energy (defined by total length) of the braids, by comparing the final equilibrium state with the initial configuration.

4.6.1 Efficacy Analysis

Table 4.1 lists the results of our experiments in computing $\alpha = C_{min}/C(0)$. It has been found that α decreases with increasing the initial number of crossings $C(0)$ but quickly approaches a limiting value. In §5, we have calculated an upper bound for α using the algebraic method of reduction as the initial crossing number $C(0)$ gets large. Further investigation shows that if $C(0) = 10n$ the resultant α is virtually at the limiting value, so that it is this initial length that was chosen for this simulation since computation time is a very real issue here. Given a value for n and a minimizing method, we generated a statistical ensemble of 1000 braids, with the same number of crossings $C(0)$ and number of strings n but otherwise randomly embedded, and we evolved them in time as long as the equilibrium constraint, described in the previous sections, was satisfied. We then computed a distribution of reduction ratios α, which turned out to be a Gaussian distribution. In Table 4.1 are reported the mean values obtained from this analysis, with an error of one standard deviation.

Except for the 3-braid, the average values in Table 4.1 suggest that the crossing number force is by far the best, among the methods analyzed, in reducing the crossing number. This result was somewhat expected because of the way this force was derived. However, it is still worth noticing that it produces reduction ratios at least 15% smaller than the other methods and even 30% smaller than the curvature elastic force. It is interesting that the heuristic algebraic method lies roughly between the two elastic approaches with the constrained elastic being the clear winner.

In Fig. 4.1, for a better comparison, we show the average reduction ratios α as a function of the number of string n. On the left, the results obtained by the heuristic algebraic algorithm are plotted with their correspondent bounds (column 2 and 3 in Table 4.1). On the right, ratios from constrained and curvature elastic relaxation are displayed with ratios from crossing number force relaxation (column 4, 5 and 6 in Table 4.1), all with error bars. All the curves exhibit a steep growth followed by a slowly increasing phase, thus suggesting an overall logarithmic behavior. In one case, namely the curvature elastic relaxation (diamond in the right panel of Fig. 4.1), we

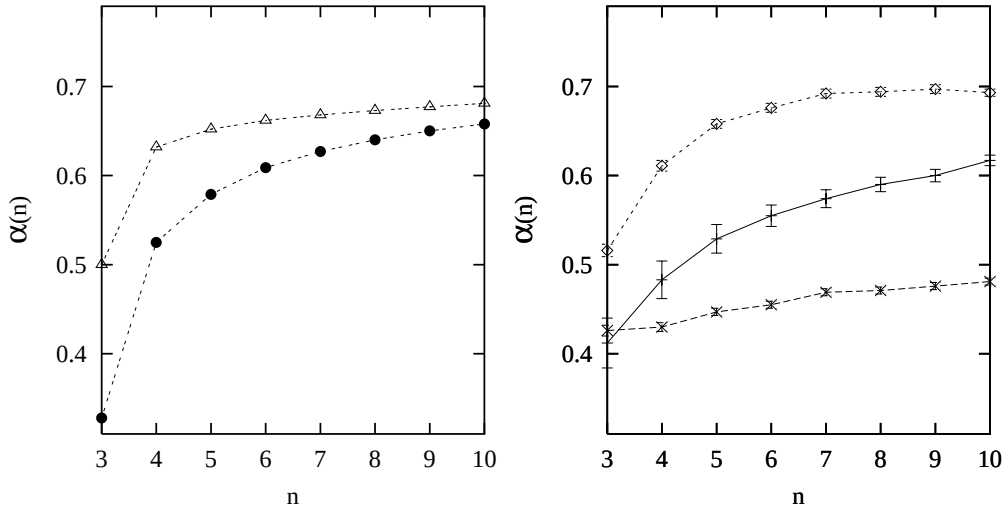


Figure 4.1: Reduction ratios $\alpha = C_{min}/C(0)$ as a function of the number of strings n. On the left, we display ratios obtained by using the algebraic algorithm ($\bullet$) and their correspondent bounds ($\triangle$); on the right, ratios are from numerical experiments with curvature elastic force ($\diamond$), constrained elastic force ($+$) and crossing number force ($\times$). For the data, refer to Table 4.1.

observe a saturation of the reduction ratio to a limiting value $\alpha_c \sim 0.69$, even if we cannot conclude this is an asymptotic value due to the relatively small maximum number of strings ($n = 10$) we have achieved in this simulation.

4.6.2 Efficiency Analysis

Having compared by how much we may shorten an average braid A, we ask how long this will take for the various methods. We will answer this by giving the complexities of all the methods and comparing them by this and their actual relative running times. The algebraic method, as stated before, has complexity $O\left(L(A)^2\right)$ independent of n. For each of the forces, we must compute and apply the force for each point on the braid. For the repulsive force we need only compute it for the points on the same level for all other strings. If we use b points per crossing and string for the simulation ($nbL(A)$ points in total), the complexity per time step is clearly $O\left(n^2 bL(A)\right)$. The number of time steps required depends upon our equilibrium condition. As described above we compute the maximum distance moved by any point on the braid and we terminate the simulation if this is less than the equilibrium parameter η (for practical purposes a maximum number of time steps must, in general, also be imposed for certain awkward cases). It would seem intuitive that if η is chosen optimally, the number of steps required would be of order n. Thus giving the optimal model a complexity of $O\left(n^3 bL(A)\right)$.

In spite of this, there seems to be no general method to estimate an optimal η and so we are not able to obtain the optimal complexity in practice. The constrained elastic force and the repulsive force vary linearly in the coordinates of the points and the difference at any time step between the coordinates is exactly the force (see our choices of parameters in equations 4.32 and 4.33). Thus the number of time steps increases linearly with $1/\eta$ giving it a complexity of $O\left(n^2 bL(A)\eta^{-1}\right)$.

In the case of the curvature elastic relaxation, the force acting on any point of a certain string depends on all the points belonging to the same string. For what concerns the stopping parameter η, dimensional analysis suggests a dependence as before, namely on η^{-1}. Thus, the complexity for the global elastic force is $O\left(nb^2 L(A)^2\eta^{-1}\right)$. Since this force is always implemented together with the repulsive force, whose complexity scales with n^2, the total complexity is identical to that of the constrained elastic energy.

The crossing force is calculated as a sum over all other strings, that is that the complexity of

each time step is $O\left(n^2b^2L(A)^2\right)$. The crossing force has two additional parameters to η, namely ϵ, the infinitesimal parameter introduced in 4.16 to avoid singularities, and R, the maximum square distance that the sum of the points on any horizontal plane are allowed to have. From the form of the force in equation 4.25 we see that it is of order $\lambda r^{-1}\epsilon^{-1}$. The results presented in this paper have been found by using $\lambda \sim \epsilon^0$ and $\epsilon = 0.05$. Clearly ϵ too close to zero is not acceptable because of the force blowup at $\epsilon = 0$. Too large ϵ is also unacceptable since the force becomes too strong and the points move too far at each time step to preserve continuity of the braid string; this transition seems to occur when ϵ becomes greater than 0.5. As far as we have investigated the choice of free parameters, these values for λ and ϵ can be assumed as optimal values. Therefore, acceptable final states are reachable in $\sim \eta^{-1}\epsilon^{-1}$ time steps giving the algorithm a total time complexity of $O\left(n^2b^2L(A)^2\epsilon^{-1}\eta^{-1}\right)$. The complexity seems to be independent of R.

We note in passing that both for the curvature elastic force and the crossing number force we actually used a grid of $N = 200$ evenly spaced points along each string: in this case, the factor $bL(A)$ simply must be read as N in the complexity estimate.

It's worth stressing that the actual amount of computation time required for the minimum crossing force is the largest. The algebraic algorithm is much faster than the elastic energy simulations in practice even though the parameter b can essentially be regarded as constant. The reason for this seems to be that the average case complexity for the heuristic is very close to linear.

4.6.3 Energy Analysis

As a final step in our comparison, we present results concerning the energy of the final braid. We limit ourselves to the three relaxing methods, since any energy estimate depends on the actual configuration of the braid in the embedding space. In fact, we took the total length of the braid to be its energy. This choice, besides being the simplest and most natural, has the advantage to give us a qualitative idea of the physical configuration of the final braid. Consider the braid $\sigma_1\sigma_2\sigma_1^{-1}\sigma_2\sigma_1^{-1}\sigma_1\sigma_2^{-1}$. In Fig. 4.2, we display the initial embedding and the final configuration of this braid after it has been relaxed by our forces. We note great similarity between the elastic energy approaches and a marked difference between the elastic and crossing number forces. The crossing number force, while driving the strings outward (to greater total braid energy) achieves a more balanced braid. There is a curious feature in the final configuration of a braid relaxed under the crossing number force which can be seen in the figure. The parts of the braid which are close (vertically) to a crossing are closer to the vertical center of the braid than the rest of the string. The elastic forces, by construction, draw the braid in on itself and thus create a braid of lower energy which sometimes results in trapping crossings; this is the main reason why the elastic approach will not, in general, actually reach the minimal length of the braid. Fig. 4.2 displays the same four pictures for the braid $\sigma_1\sigma_2^{-1}\sigma_1^{-1}\sigma_2^{-1}\sigma_1$ also in order to make more apparent the features of the forces.

Table 4.2 gives the mean values of the final energy for the constrained and curvature elastic force as well as the minimum crossing force, obtained from the same set of data above. For the sake of comparison, the mean initial energy of each ensemble of randomly generated braids is given in the second column. Since the total length depends upon the number of strings, we have divided the final energy by the number of strings. Note that we have defined a geometrical braid to lie between the planes $z = 0$ and $z = 1$ so that this is already normalized. Thus the minimum possibly energy of any n-braid is 1.

As expected, the energy is systematically increased by the crossing number force (see in Fig. 4.3 crosses versus bullets). Besides, the fact that we confine the physical braid in a cylinder in order to prevent the occurrence of singularities, imposes a limit on the final energy, which more or less oscillates about a fixed value. On the contrary, the constrained and curvature elastic forces reduce the energetic content of the braid. While the curvature force reduces the energy more effectively than the local, the descent of energy as a function of n is steeper for the local energy (see Fig. 4.3) and so we may expect an intersection of these methods at about $n = 13$. Note that the mean initial energy shows a slight dependence on n, due to the embedding procedure we use.

Once a minimal configuration has been reached, one may wish to know whether it is a local or a global minimum. This is a very difficult question to answer and we have not endeavored to do so. However, because of the constraint that the endpoints of the braid may not move, we may safely say that the number of distinct local minima is finite and low.

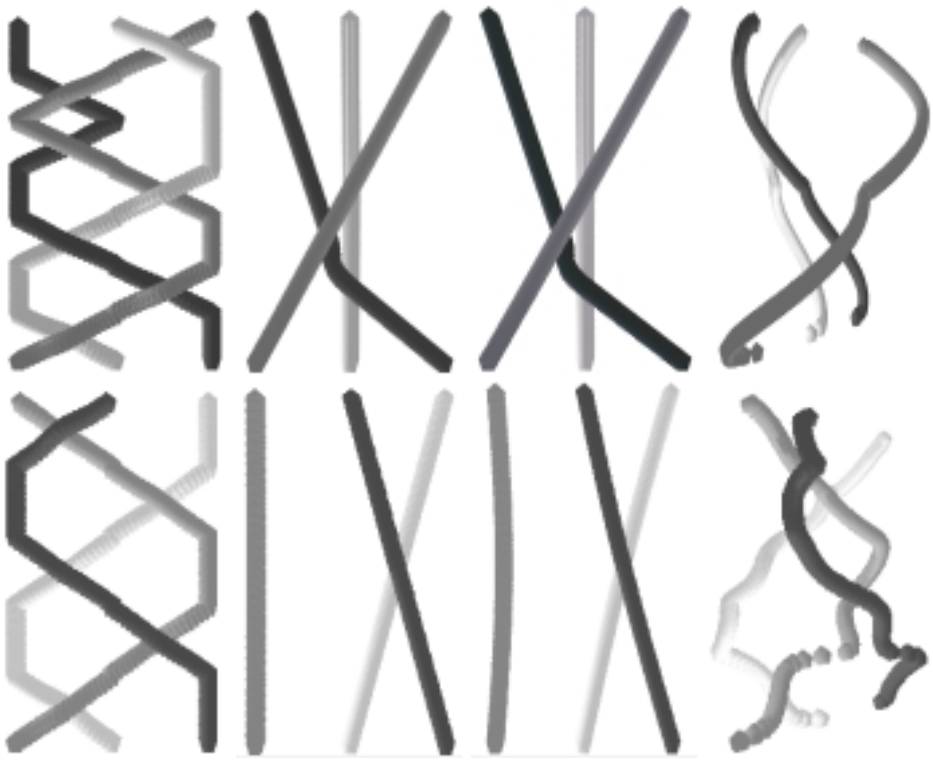


Figure 4.2: This figure gives the initial embedding and the final configuration after the constrained elastic, curvature elastic and crossing number forces have been applied for the two braids $\sigma_1\sigma_2\sigma_1^{-1}\sigma_2\sigma_1^{-1}\sigma_1\sigma_2^{-1}$ and $\sigma_1\sigma_2^{-1}\sigma_1^{-1}\sigma_2^{-1}\sigma_1$ respectively. We see great similarity between the elastic approaches but substantial differences between them and the crossing number force. Note that the diagrams for the crossing number force have been rotated by $\pi/4$ to make more apparent the curious deformations of the strings. (The images were generated using BraidLink, a software program written by the author.)

Table 4.2: Total length per string.

n	Initial Energy	Constrained Elastic	Curvature Elastic	Crossing Force
3	1.796 ± 0.001	1.793 ± 0.003	1.323 ± 0.010	2.42 ± 0.03
4	1.858 ± 0.001	1.690 ± 0.002	1.333 ± 0.006	2.28 ± 0.02
5	1.911 ± 0.001	1.660 ± 0.001	1.314 ± 0.004	2.31 ± 0.02
6	1.960 ± 0.001	1.608 ± 0.001	1.298 ± 0.004	2.38 ± 0.02
7	2.000 ± 0.001	1.550 ± 0.001	1.292 ± 0.003	2.41 ± 0.02
8	2.032 ± 0.001	1.501 ± 0.001	1.288 ± 0.003	2.35 ± 0.02
9	2.054 ± 0.001	1.448 ± 0.001	1.282 ± 0.003	2.36 ± 0.02
10	2.076 ± 0.001	1.411 ± 0.001	1.279 ± 0.003	2.41 ± 0.02

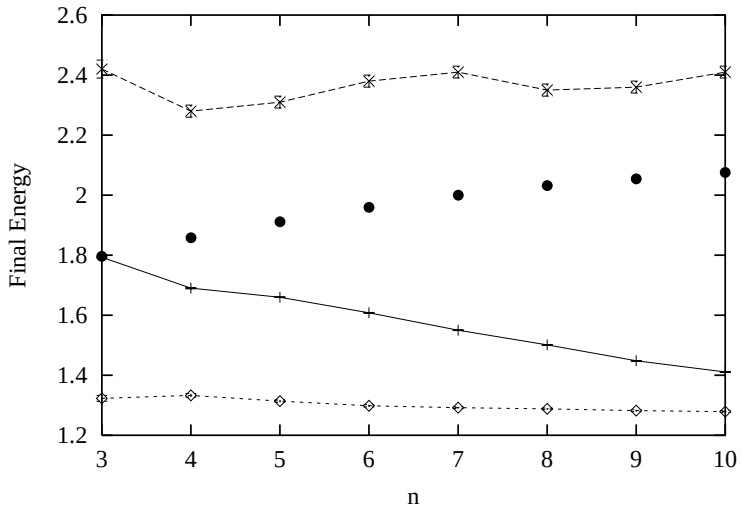


Figure 4.3: The total length per string is displayed here as a function of the number of strings for the curvature ($\diamond$) and constrained ($+$) elastic energies and the crossing number force ($\times$) as well as the initial energy before relaxation ($\bullet$).

4.7 Conclusions

We have investigated, by means of computer simulation, different methods to reduce the crossing number of a braid over its equivalence class. As a group theoretical question, this problem is difficult (if the minimum is to be found [125]) but can be profitably approached using a heuristic algorithm presented above. A braid can also be regarded as a topological object divested of this algebraic approach. Here the strings may move (except the endpoints) in the embedding manifold without crossing each other. For algorithmic purposes a systematic way to move the strings must be found based on certain principles. Two of our approaches center on a physical model of the strings as elastic strings made of flexible material. Elasticity may be modeled using a nearest neighbor or curvature approach, both of which were investigated. Another way to systematically move the strings is to construct a force not based on a physical idea but by using the crossing number (as an integral) as a potential in a Lagrangian. This last approach has proved to be the most successful in terms of finding the shortest braid, on average. It is however the most time consuming method. The algebraic approach, while only third (out of the four methods) in reduction efficacy, is the fastest by far.

In many applications, the braid is already an embedded topological object and not an element of the braid group. Here the two energy methods find their application as they are the only physically relevant methods. In solar physics, for example, the magnetic field lines may be modeled as braids. These seem to behave as elastic configurations over time. It must be mentioned that the endpoints of these braids do move but in a random fashion. Research about this added complication is in progress. In physical applications, we are most concerned about the energy of a braided configuration and the elastic model seems to be the most realistic for a variety of applications. While the constrained approach is more successful in terms of crossing number, the curvature fares better in an energetic sense.

What has clearly emerged from the discussion above is that minimum energy and minimum crossing number for braids are different things. While reducing energy does also reduce crossing number, reducing crossing number does not necessarily reduce energy and crossing number may be reduced much further after the minimum energy configuration has been reached. Thus, it is clear that the elastic approaches terminate in a *local* minimum as far as the equivalence class of the initial braid is concerned. From the point of view of ideal knot theory, this result is significant because it has often been suggested that by reducing some form of knot energy, one may find a knot which is particularly simple over its equivalence class. Whether this measure of simplicity coincides with minimum crossing number over all possible projections (the traditional measure of

simplicity used by knot tabulators such as Tait) has given rise to some debate for which our result provides additional fuel.

We conclude our investigation by saying that the algebraic method provides a useful minimization approach for purely group theoretical work, the crossing force is the best approach when one wishes to find an especially short braid (and is not bound to a purely group theoretical framework) and the curvature elastic energy is the best scheme to minimize elastic braid energy, i.e. total string length.

Chapter 5

Knotation and Braiding a Knot

We consider the notion of a tangle and analyze the operations which are possible on it. We use tangles in constructing a new notation for knots based on Conway's knot notation. This new notation has several advantages over existing notations. All the basic properties of the notation and algorithms to retrieve simple knot information are discussed. Procedures for putting a knot into our notation are also given. Finally, polynomial-time algorithms, which do not rely on topological deformation, are described which produce a plait and a closed braid which are isotopic to any knot given in our notation.

First, we review the notion of tangles and investigate their classification. We shall then introduce the new notation, prove that all knots may be represented by it, give an algorithm to place a given knot into this notation and present a traversal algorithm which will calculate certain features of the knot. An algorithm is then given to obtain a plait and a braid, the closures of which are a given knot in the new notation.

5.1 Tangles

5.1.1 Definition and Partition

Consider the 3-ball B^3 and choose $2n$ points on its surface, which is the 2-sphere S^2, and call the set of these points P. Attach n polygonal curves to the $2n$ points such that: (i) each curve intersects S^2 in exactly 2 points in P, which are its endpoints, (ii) exactly one curve may begin or end at any one point in P and (iii) no curve may intersect another. If the set of these curves is T, then we will call the set $\left(B^3, T\right)$ an *n-tangle*. In particular, we will focus on 2-tangles and so whenever we skip the n, it will be understood that we mean $n = 2$. Note that our requirement that the curves be polygonal excludes any wild tangles, where wild is to be understood in the usual knot theory sense. Two tangles are called *equal* if they are isotopic without moving the points in P.

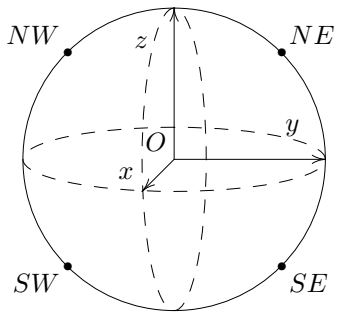


Figure 5.1: The 3-ball and the four points on its surface which form the endpoints of the two polygonal curves necessary to define a tangle.

A tangle can be visualized readily by choosing the four points (named according to the cardinal

points of the compass)

$$NE = \left(0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right), \qquad NW = \left(0, -\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) \tag{5.1}$$

$$SE = \left(0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right), \qquad SW = \left(0, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right) \tag{5.2}$$

on the unit sphere, which will be our canonical B^3, see figure 5.1. Even though tangles are, by definition, three dimensional objects, we will work with their projection onto the two dimensional plane as if the projection is the tangle. The fact that a projection in which there are at worst double points always exists for a tangle follows from the corresponding theorem about knots.

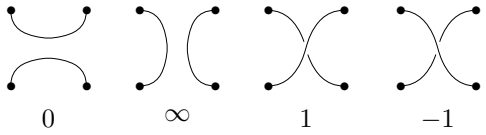


Figure 5.2: The elementary tangles.

We shall find it convenient to partition the set of all possible tangles into a few categories: elementary, integral, fractional, rational and irrational. The simplest are the *elementary tangles*, of which there are four. These are best introduced by displaying them in figure 5.2. Note that we have not drawn B^3, it should however be understood to be present. The reason for naming them as they have been will become apparent later on. Note that the literature disagrees on which of the two tangles ± 1 is to have the minus sign, this is a matter of convention and has no serious consequences (we follow the convention introduced by Conway).

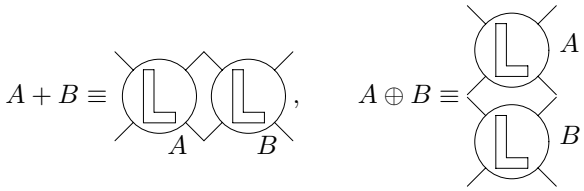


Figure 5.3: Tangle addition.

The other types of tangles can be most readily defined in terms of combining the elementary ones in some way. To do this, we shall define two ways of adding tangles. Following Conway, we denote a general tangle by an "L" shaped symbol within the 3-ball and we also sketch the ends of the two curves by which tangles may be attached to one another. In this way, we define the horizontal sum $+$ and the vertical sum $\oplus$ in figure 5.3.

In what follows, we shall use a superscript to denote of which type a particular tangle t is; for example an elementary tangle t would be denoted by $t^{(e)}$. An *integral tangle* $t^{(i)}$ and a *fractional tangle* $t^{(f)}$ will be defined in terms of the elementary tangles ± 1 by

$$t^{(i)} = \underbrace{1 + 1 + \cdots + 1}_{t \text{ factors}} \tag{5.3}$$

$$t^{(f)} = \underbrace{1 \oplus 1 \oplus \cdots \oplus 1}_{t \text{ factors}} \tag{5.4}$$

The negative versions are, of course, the sums of -1 tangles instead of 1 tangles. A *rational tangle* $t^{(r)}$ can then be defined in terms of a sum of integral and fractional tangles. The definition of the sum differs if the number of tangles j in the sum is even or odd, this is because the definition requires an alternate sum between integral and fractional (and the two methods of addition) which always *ends* in an integral tangle being added. This is because the set of rational tangles may

be classified if this restriction is imposed; the classification scheme is outlined in the next section. The integral tangles, including the last, may be zero and the fractional tangles may be infinite. If any component tangles are 0 or ∞ though, they may be removed from the sum and the terms immediately preceding and following the removed term may be added together to shorten the sum, while preserving isotopy.

$$t^{(r)} = \underbrace{a^{(i)} \oplus b^{(f)} + c^{(i)} \oplus d^{(f)} + \cdots + z^{(i)}}_{j \text{ odd}} \tag{5.5}$$

$$t^{(r)} = \underbrace{a^{(f)} + b^{(i)} \oplus c^{(f)} + d^{(i)} \oplus \cdots + z^{(i)}}_{j \text{ even}} \tag{5.6}$$

Note that the set of elementary tangles is a subset of both the integral and fractional tangle sets which are subsets of the rational tangle set. We shall call any tangle which is not rational, *irrational*.

5.1.2 Classification of Tangles

We may denote a rational tangle by giving its integral and fractional factors in order. Thus a sequence of integers $t^{(r)} = (a_1, a_2, \ldots, a_i)$ defines any rational tangle. Note again that the identity of the tangle factors is decided by requiring the last in the sequence to be integral. Given a rational tangle $t^{(r)} = (a_1, a_2, \cdots, a_i)$, we may associate with it an extended rational number $E(t^{(r)}) = \alpha/\beta$, where α and β are integers including zero. We say an extended rational number because this allows for $1/0 = \infty$, the inclusion of which extends the rational numbers. We calculate $E\left(t^{(r)}\right)$ by the continued fraction (the + signs are arithmetic additions and not tangle additions)

$$E\left(t^{(r)}\right) = a_i + \cfrac{1}{a_{i-1} + \cfrac{1}{a_{i-2} \cdots + \cfrac{1}{a_1}}} \tag{5.7}$$

Conway [50] was able to deduce that two rational tangles are equal if and only if the associated extended rational numbers were equal, this is called Conway's Basic Theorem. The first published proof may be found in [39] but a more intuitive proof was given by Goldman and Kauffman [71]. Thus Conway's Basic Theorem classifies rational tangles in a simple algorithmic manner.

In particular, the fractions associated with the elementary tangles are their numerical names: 0, ± 1 and ∞. The fraction for an integral tangle $t^{(i)}$ is $t^{(i)}$ and for a fractional tangle $t^{(f)}$ is $1/t^{(f)}$. It is clear now why these tangles were named as they were. This concludes our review of previous work on tangles and the rest of the chapter is new work.

By equation 5.7 is easy to calculate the fraction associated with a given rational tangle. Given a fraction, it is also possible to decompose it into appropriate factors, thereby constructing the rational tangle associated with it. Euclid's algorithm will accomplish this.

5.2 Knot Notation

Tangles were invented in an effort classify knots (they may be used to classify two-bridge knots via the correspondence with the extended rational numbers [116]) and so we must have a method to combine tangles into knots. Conway [50] showed that any knot may be obtained by substituting several rational tangles into the vertices of basic polyhedra. A *polyhedron*, in the sense of Conway, is an edge-connected 4-valent planar map and it is *basic* if, in addition, no region (including the infinite region) has just two vertices. Conway constructs the 8 different basic polyhedra necessary to denote all prime knots up to and including 11 crossings. The beauty of using the basic polyhedra is that small knots may be named quite efficiently, that is one gives the basic polyhedron and the tangle fractions to be substituted. However it can be quite a chore to construct the Conway name of a large knot. The next section will introduce our new knot notation.

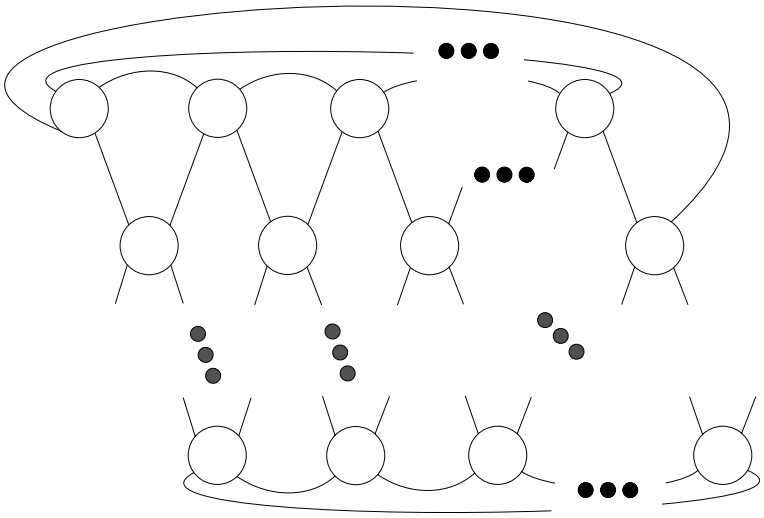


Figure 5.4: The universal polyhedron

5.2.1 The Universal Polyhedron

Consider the basic polyhedron $P(i,j)$ shown in figure 5.4; we will call it the *universal polyhedron*. It is a prototype for a knot projection. The circles will be called *vertices* and the lines connecting them *edges*. The vertices are arranged into i rows of j vertices each. Each vertex can thus be labeled by its row and column index. While $P(i,j)$ denotes the whole polyhedron and specifies the number of rows and columns, p_{kl} specifies a particular vertex in row k and column l. In what follows, we will substitute rational tangles into the vertices to yield a knot projection. Since a rational tangle may be specified by a single extended rational number, p_{kl} takes an extended rational number value. By substituting rational tangles into all vertices of a given polyhedron, we obtain a knot projection of some tame knot. This can be completely specified by giving all p_{kl} a value, which may be arranged into a matrix form,

$$P(i,j) = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1j} \\ p_{21} & p_{22} & \cdots & p_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ p_{i1} & p_{i2} & \cdots & p_{ij} \end{pmatrix} \tag{5.8}$$

Giving a matrix with extended rational number valued entries completely specifies a knot projection (the parameters i and j of the polyhedron are just the number of rows and columns, respectively, of this matrix). Since this is true for rational tangles, it is true for any subset of the rational tangles, in particular the elementary tangles. Thus if all p_{kl} take a value from the set $\mathcal{E} = \{0, -1, +1, \infty\}$, the result is also a knot projection.

We wish to find this matrix notation for a given knot projection. Since this is an algorithmic question, we must ask in what fashion the knot is already given. Generally a knot is given by one of its projections onto the plane. A knot projection is characterized by n double points and $2n$ arcs connecting them. The information we must encode into our notation is which type of double point each of the n points are and which other points they are connected to.

Algorithm 5.2.1 *Input: A knot projection with n crossings. Output: A knot projection given in our matrix notation.*

1. Circle each double point in the knot projection and name them by letters in the alphabet $A, B, \cdots$. The naming may start at any point on the knot but the order should be in the order the crossings are encountered when traversing the knot in the direction of its orientation.

2. For each double point, number the intersections of the knot with the circle drawn in step 1 from one to four in clockwise order starting at an arbitrary point. There are 4^n such numberings.

3. The connections may now be written down from the knot projection, for example "$A1 \to B2$." For n crossings, there will be $2n$ such connection rules.

4. Insert double point A into vertex $(1, 1)$ such that point $A1$ is at the top left of the vertex.

5. Insert the other double points into the polyhedron in order making all the necessary connections, as enumerated in step 3, between the current point and all the previously inserted points. The connections are to be made by using the intrinsic connections of the polyhedron and the 0 and ∞ tangles. Increase the number of rows and columns in the polyhedron dynamically as this becomes necessary.

6. When the final point is added and all connections have been made, the algorithm is complete.

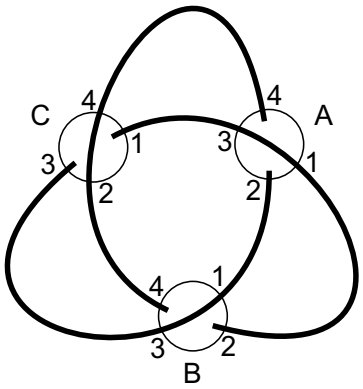


Figure 5.5: Naming and labeling the points in a projection of the trefoil knot.

To illustrate this algorithm, we shall find the knotation for the trefoil knot. We have drawn the standard projection of the trefoil knot in figure 5.5 and have circled the double points, named them and numbered the four intersections between the ± 1 tangles which are the double points and the circles. This completes steps one and two of the algorithm. In step three, we must write down how the points are to be connected,

$$A1 \to B2 \qquad A4 \to C4 \tag{5.9}$$

$$A2 \to B1 \qquad B3 \to C3 \tag{5.10}$$

$$A3 \to C1 \qquad B4 \to C2 \tag{5.11}$$

Inserting double point A into vertex $(1, 1)$ in step four requires us to insert a -1 tangle into the $(1, 1)$ position in the matrix. We must now add double point B. Note that we have numbered B in such a way that $A2 \to B1$ which is an intrinsic connection in the polyhedron if we put B in vertex $(1, 2)$. Since we also have $A1 \to B2$, we would like the final polyhedron to have only two columns since then this connection too is intrinsic to the polyhedron. Since these are the only connections between A and B, we now have to add C. Note that we have numbered C in such a way that $A3 \to C1$ which lets us place C underneath A. Since we also have $B4 \to C2$, we accommodate both rules by the intrinsic polyhedron connections by putting C into vertex $(2, 1)$. The last two rules $A4 \to C4$ and $B3 \to C3$ can be incorporated by placing an ∞ tangle in vertex $(2, 2)$ and requiring the polyhedron to have two rows. This completes the insertion of all double points, completes all the connections and fixes the rows and columns of the polyhedron. All vertices are filled and thus the algorithm is complete. The polyhedron with the inserted tangles is shown in figure 5.6 and the result is

$$\text{trefoil} = \begin{pmatrix} -1 & -1 \\ -1 & \infty \end{pmatrix} \tag{5.12}$$

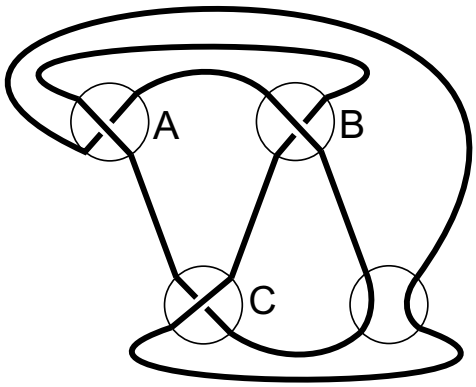


Figure 5.6: The trefoil knot in $P(2,2)$

An advantage of the above algorithm is that the number of rows and columns is only increased when necessary. The disadvantage is that it is not possible to say, *a priori* how many rows and columns a knot of n crossings will need. The most crucial step in the algorithm is step two. If the intersection points are numbered with foresight, then most connections can be made by making use of the intrinsic connections in the polyhedron and will not require the addition of 0 or ∞ tangles. Arguing like this, one might be lead to believe that it should in general be possible to knotate an n crossing knot in a polyhedron $P(i,j)$ where $i = j = \lceil \sqrt{n} \rceil$, i.e. the least integer greater than $\sqrt{n}$, we can not prove this however. Having presented an algorithm and an example, we give a proof that any knot can be represented in some $P(i,j)$ using only elementary tangles.

Theorem 5.2.2 *Every regular projection of any knot may be represented by the universal polyhedron $P(i,j)$ for some i and j all the vertices of which contain elementary tangles.*

Proof. A regular projection of a knot is characterized by a finite number n of double points and $2n$ arcs which connect the double points in a specific manner. For sufficiently large i and j, the polyhedron $P(i,j)$ can accommodate all double points in the form of ± 1 tangles and can achieve the desired connection of these by placement of 0 and ∞ tangles into it. This is obvious because the 0 and ∞ tangles represent horizontal and vertical connectors in the polyhedron. Because this connection may be achieved without ± 1 tangles, it is clear that no further components, with the possible exception of unknots, are created. Thus what remains to be shown is that no unwanted unknots will be created.

There are ij vertices and $2ij$ edges connecting them in the empty polyhedron $P(i,j)$. Eliminating one vertex by a 0 or ∞ tangle, eliminates two edges. Apart from the ± 1 tangles of which there are n, the final polyhedron will contain $ij - n$ tangles of type 0 and ∞ which will have eliminated $2(ij-n)$ edges from the original polyhedron, leaving exactly $2n$ edges which are needed to connect the double points. Thus there is no extra edge left over which could possibly form an extra component. Therefore any knot may be represented using the basic polyhedron $P(i,j)$ and elementary tangles. □

In the section 5.4, we give a set of equations relating our notation to Conway's so that a knot given in either notation can be immediately translated to the other. We also present an algorithm with which a knot given in our notation using non-elementary integral tangles may be transformed into a knot using only elementary tangles.

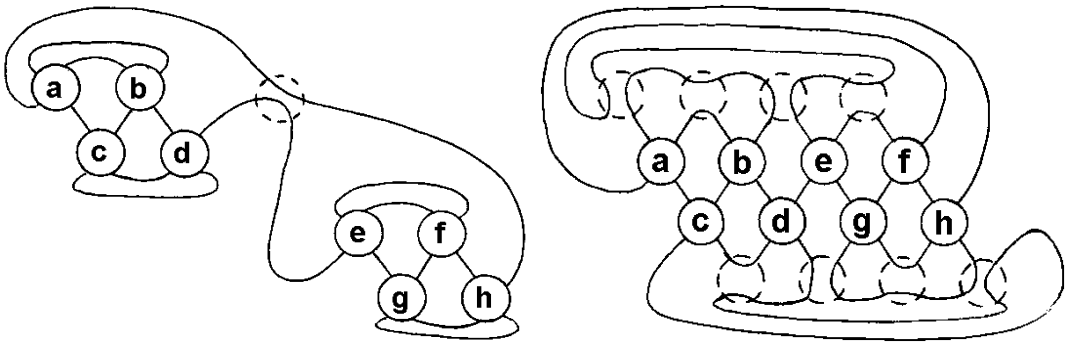
We have been discussing unoriented knots but the orientation must also be encoded in the notation if required. Each elementary tangle is composed of two strings which may have two orientations each. Therefore each elementary tangle has four different possible orientations. It follows from theorem 5.2.2 that any oriented knot may be represented by the polyhedron $P(i,j)$ in which the oriented elementary tangles are inserted into the vertices. While every matrix with elementary tangles in all its entries denotes a valid knot, not all matrices with oriented elementary tangles denotes a valid oriented knot because we have to require the orientations of the tangles to be compatible. Whether a given matrix does represent a valid knot may be decided by the knot traversal algorithm presented in the next section.

5.2.2 Basic properties

We will now assume that the knot of interest is given in our matrix notation with $p_{kl} \in \mathcal{E}$. A few basic properties of the notation need to be enumerated before the notation becomes useful in dealing with knots. These properties include the behavior under knot addition, calculating the number of components in a knot and calculating the number of regions in the projection.

Two knots may be combined into a single knot by addition. In adding, each knot is cut at a single point and the ends are spliced together. We find that given two knots A in $P(i,j)$ and B in $P(i',j')$ such that $j' \geq j$, the sum $C = A\#B$ is given by

$$C = \begin{pmatrix} \infty & 0 & \infty & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{11} & a_{12} & a_{13} & \cdots & a_{1i} & b_{11} & b_{12} & b_{13} & \cdots & b_{1i'} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2i} & b_{21} & b_{22} & b_{23} & \cdots & b_{2i'} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{j1} & a_{j2} & a_{j3} & \cdots & a_{ji} & b_{j1} & b_{j2} & b_{j3} & \cdots & b_{ji'} \\ 0 & 0 & 0 & \cdots & 0 & b_{j+1\,1} & b_{j+1\,2} & b_{j+1\,3} & \cdots & b_{j+1\,i'} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & b_{j'1} & b_{j'2} & b_{j'3} & \cdots & b_{j'i'} \\ 0 & \infty & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \tag{5.13}$$



(a) (b)

Figure 5.7: Addition of two knots in our notation.

This addition formula is the straightforward consequence of the geometrical procedure of splicing two polyhedra together and then making them both fit into another larger one. We illustrate this procedure for two $P(2,2)$ polyhedra in figure 5.7. As can be seen in figure 5.7, this method chooses a particular cutting point for each polyhedron. It can be shown that the operation of knot addition # is independent of the cutting point. This is true only within a component of a knot. If a knot has more than one component, the method of adding described by equation 5.13 is not general but makes a specific choice. Because such additions rely on the particular structure of the specific knots to be added, such a formulation can not be made in general.

This new notation can be readily used in calculating some invariants of the knot. For example, to calculate a polynomial invariant for which we have a state model, we simply replace each ± 1 tangle by the 0 or ∞ tangles in all possible ways to yield all possible states of the knot. If a knot has n double points, this means 2^n states. We associate an algebraic factor with the way in which this replacement is made and then multiply it by an algebraic factor depending on the number of unknots left (since there are no double points left, this is equivalent to the number of regions in the resultant projection). All these contributions are added and yield a polynomial invariant of the knot. The key is to be able to calculate the number of regions and components of a knot given its matrix.

The number of regions into which the knot projection partitions the plane is important in a few applications such as the calculation of polynomial invariants, as stated above, and also in the braiding algorithm which follows. Each vertex has exactly one region lying to its right in the

polyhedron and thus we may label all these regions by the row and column indices of the associated vertex. Only two regions are not indexed by this method, these are the two regions with j vertices directly on the top and bottom of the matrix construction. These will be labeled by the pairs $(0,1)$ and $(j+1,1)$. Thus the regions may also be represented by a matrix. If the entries r_{kl} are made to take integer values we may count the number of regions by the following algorithm.

Algorithm 5.2.3 *Input: A matrix describing a knot in our notation. Output: A matrix describing the regions of the knot. Each element of the matrix receives a label from 1 to R, the number of regions. This gives complete information about which regions of the polyhedron are connected and how many there are.*

1. Begin at the top left of vertex $(1,1)$ and follow the boundary downwards, as for counting regions, the orientation of the knot does not matter. Mark the region $(0,1)$ with a 1, the current marker, in the region matrix.
2. In following the boundary, one will come to vertex $(1,1)$; we assess its value and continue. If we stay in the same region of the *polyhedron* we continue, if we enter a new region of the polyhedron, then this new region of the polyhedron belongs to the same region of the *knot* as the previous one and thus we mark it with the current marker in the region matrix. The whole issue at hand is that the regions of the polyhedron are known while we wish to gain knowledge of the regions of the knot.
3. We continue to follow the boundary until we reach the point of origin.
4. We search the matrix for an unmarked region. If there exist unmarked regions, we increment our current marker and choose one of the regions as our new starting region and choose a point upon its boundary as our new starting point. Then, we repeat the algorithm from step 1, marking the region with the current marker.
5. Once no unmarked region of the polyhedron exists, the algorithm is finished. The largest marker used in the matrix which we have obtained is clearly the number of regions of the knot. Furthermore, since all connected regions are labeled with the same marker, we have a complete knowledge of which regions of the polyhedron belong to the same region of the knot.

The algorithm considers each vertex exactly twice and moves and marks accordingly. Therefore the complexity is $O(n)$. An algorithm to find the number of components in a knot is similar but differs in a few details.

Algorithm 5.2.4 *Input: A matrix describing a knot in our notation. Output: The number of components in this knot.*

1. Each vertex has four points in which the two polygonal curves intersect B^3. These are shown in figure 5.1. Start at point NW of the vertex $(1,1)$ and follow the orientation of the knot.
2. We follow the orientation and not the boundary, as in algorithm 5.2.3, marking each point as we pass it.
3. When we reach the point of origin again, we increment the component counter and look for an unmarked point.
4. If there is an unmarked point, we begin with step 1, if there is not, we are finished.

This method calculates the number of components considering each point on each vertex once, therefore the complexity is also $O(n)$. Note that a matrix of only 0 tangles contains $i+1$ unknots and a matrix composed of only ∞ tangles contains j unknots.

Clearly smaller polyhedra $P(i,j)$ may be embedded in larger ones by filling in the rest with 0 and ∞ tangles. Conversely, if the configuration of the tangles is right, we may delete rows and columns accordingly. For example, we may create an extra row at the bottom or top of the matrix containing

$$(0 \quad 0 \quad \cdots \quad 0 \quad \infty) \tag{5.14}$$

and we may add an extra column at the left or right of the matrix containing only 0 tangles. Likewise, such columns or rows may be removed without changing the knot type. Thus if a given knot can be expressed in the polyhedron $P(i, j)$ it can also be expressed in any polyhedron $P(i', j')$ for which $i' \geq i$ and $j' \geq j$. An internal row of 0 tangles splits the polyhedron into two parts each described by the matrix above and below the row of zeros. Thus if two knots should be described in a single diagram without touching, this is a way in which this may be done.

5.3 Braiding a Knot

Having constructed a new notation for knots, we wish to solve the problem of how to extract a closed braid from the matrix which is isotopic to the knot described by the matrix. A few algorithms have been constructed in the past, which convert a knot into a closed braid but they are difficult to implement because they depend upon topological deformation of the knot projection [97] [22]. The best known algorithms have been implemented [151] [162] and have complexity $O(n^2)$. We shall present an algorithm which achieves the conversion with complexity $O(n)$, increases the number of crossings only in a few cases (and then only by a few crossings) and uses a linearly bounded number of strings. There exists no algorithm to calculate the number of strings which are at least necessary to describe a specific knot — the braid index of the knot. Because of this, it is not possible to say how close to the minimum the number of strings used by our algorithm is. The number of crossings is sometimes increased because it has been found that there are knots for which any closed braid representative has more crossings than the minimal knot diagram; the knot 5.1 in the standard tables is the simplest example of this [131]. Our algorithm is valid both for oriented and unoriented knots.

5.3.1 An Example

Alexander's theorem was proven by showing that every knot can be deformed into a form where the knot loops around an axis a finite number of times without local maxima or minima with respect to that axis. If we cut the string along the axis in one place, we obtain a braid. The gluing back of the cut constitutes the canonical closure. Thus as far as the canonical closure is concerned, the finding of an appropriate axis is the key. Having obtained a canonically closed braid which is equivalent to a knot, we may obtain a plait from it by considering the closure curves part of the braid diagram and moving them into the middle of the braid diagram. The next section gives an example of this.

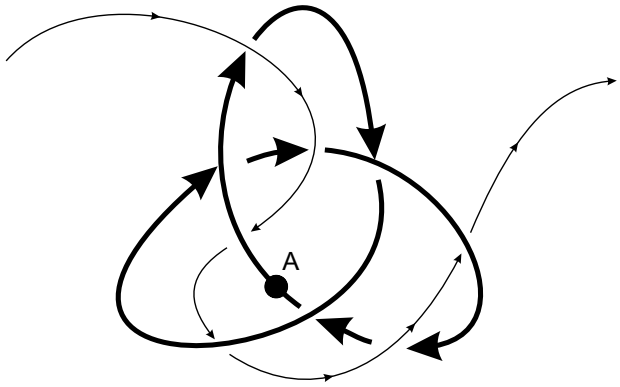


Figure 5.8: The trefoil knot with an axis for braiding it.

For the rest of this section, we are going to work through an example of our method. Consider the trefoil knot in figure 5.8. We have drawn an axis through it by the following method: (1) We drew a line through the projection of the trefoil which intersects every region of the plane at least once, (2) begins and ends in the infinite region and then (3) assigned the under and overpasses of the knot under and over the axis by traversing the knot from a random starting point (point A in the figure) while (4) assigning the passes alternately as we met the crossings of axis and knot.

Next we perform a coordinate transformation from the knot reference frame (figure 5.8) to the axis reference frame in figure 5.9 by pulling the axis straight.

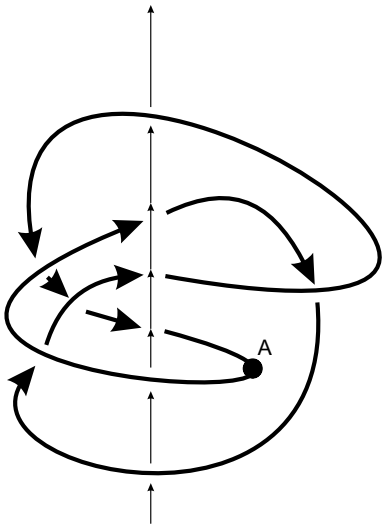


Figure 5.9: The trefoil knot as it appears after the axis has been straightened from figure 5.8. For reference the point A has been labeled here again.

We can easily observe from figure 5.9 that the axis is valid; i.e. if we traverse the knot starting at A we will travel around the axis without local maxima or minima permanently in a clockwise direction. If we now cut the knot at those points at which it overcrosses the axis and lay out the ends carefully to either side, we shall obtain the braid $\sigma_1^{-1}\sigma_2^{-1}\sigma_1^{-1}\sigma_2^{-1}$ shown in figure 5.10 (a). To get back to the trefoil from this, we perform the canonical closure which is identical to sealing the cuts made above. This is shown in figure 5.10 (b). This knot has four crossings and is ambient isotopic to the trefoil thus there is some inefficiency in our braid representation (note however that there exist knots for which the most efficient braid representation contains more crossings than their most efficient knot projection [131]). We note that we may lift the arc labeled in figure 5.10 (b) to remove one crossing. This move also removes a string and so we obtain the braid of figure 5.10 (c). This braid has two strings and three crossings, it is thus the most efficient representation of the trefoil as the trefoil must have at least this many strings and crossings. We conclude that the closure of the braid $\sigma_1^{-1}\sigma_1^{-1}\sigma_1^{-1}$ is ambient isotopic to the trefoil knot. Note that we may turn the entire figure 5.10 (c) about a vertical axis through its center and thus obtain the result that the braid $\sigma_1\sigma_1\sigma_1$ is ambient isotopic to the trefoil also; this, finally, is the well-known braid representation of the trefoil knot. This is the prototype for a general method which we shall develop below.

5.3.2 Platting a Knot

The diagram of a knot which is expressed as a closed braid may be naturally divided into two parts: the braid and the closure. The most important feature of the braid part, for our purpose, is the requirement that all strings be monotonic increasing in the vertical coordinate, that is they may only go side to side and never double back on themselves. In this light, consider turning the polyhedron $P(i,j)$ clockwise by $\pi/2$. If the polyhedron does not contain any ∞ tangles, this is already a canonically closed braid. However, in general, the polyhedron will contain ∞ tangles. Note that the rotation will make the ∞ tangles look like 0 tangles. In an effort to rid ourselves of the ∞ tangles, we take the top string in the ∞ tangle and move it all the way to the bottom of the knot diagram and move the bottom string all the way to the top. In this way, we have created two extra strings in the braid which are closed in the plait manner. If we do this for all ∞ tangles, we will have a valid braid in the center of the diagram but the closure mechanism will be a hybrid between the canonical and plait methods. In order to rectify the situation, we move the strings which are closed in a canonical manner into the center of the braid diagram, thereby creating more

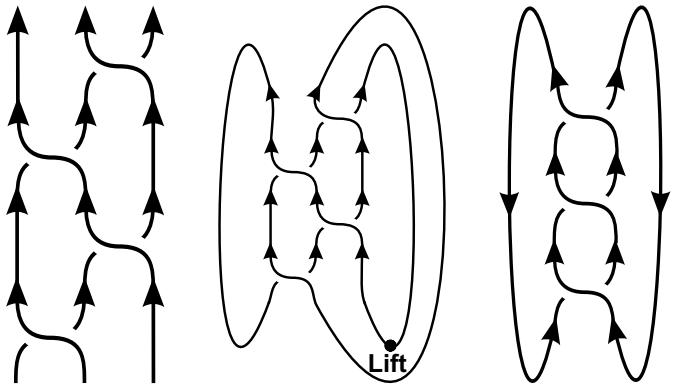


Figure 5.10: The braid which is extracted from figure 5.9 by cutting the trefoil knot at its overcrossings over the axis and laying out the ends is displayed in part (a). The closure of this braid is part (b). If we lift the arc labeled in part (b) we obtain the braid in part (c). See discussion in the text.

strings and more crossings. Once this has been done, we have a fully valid braid closed in the plait manner which is ambient isotopic to the knot we started with. Figure 5.11 shows the process of converting the unknot

$$U = \begin{pmatrix} -1 & 1 \\ \infty & -1 \end{pmatrix} \tag{5.15}$$

into the braid $\sigma_2\sigma_4^{-1}\sigma_3\sigma_4\sigma_5^{-1}\sigma_6^{-1}\sigma_4^{-1}\sigma_5^{-1}\sigma_4\sigma_6$ closed in the plait manner. This procedure is valid generally and clearly represents a readily implementable algorithm for transforming a knot given in our notation into a plait. If the original knot is given in the polyhedron $P(i,j)$ and has k tangles of type ∞, then the number of strings required in the plait is $2(i+k+1)$ but the number of crossings depends upon the exact configuration.

5.3.3 Laying the Axis

As mentioned before, the transformation of a knot projection into a canonically closed braid centers around finding an appropriate axis for the string to wind around. This was the central point of Alexander's theorem which proves that such an axis may always be found. A ready method for finding an axis is given in the following algorithm.

Algorithm 5.3.1 *Input: A knot projection. Output: A knot projection with an axis around which the knot winds without local maxima or minima.*

1. Begin with enumerating the regions into which the knot projection divides the plane, suppose there are R of these.

2. Choose two arbitrary points in the infinite region and call them A and B.

3. Draw a line L connecting A and B in such a way that the line intersects every region at least once.

4. Choose a random point on each of the knot's components and traverse the knot in the direction of the orientation once for each component starting at the chosen point. While traversing label each intersection of L with the knot alternatingly with a $+$ or $-$ sign starting with $+$.

5. Interpret each $+$ crossing as an overcrossing of L over the knot and each $-$ crossing as an undercrossing of L under the knot. The line L oriented from A to B is then a valid axis.

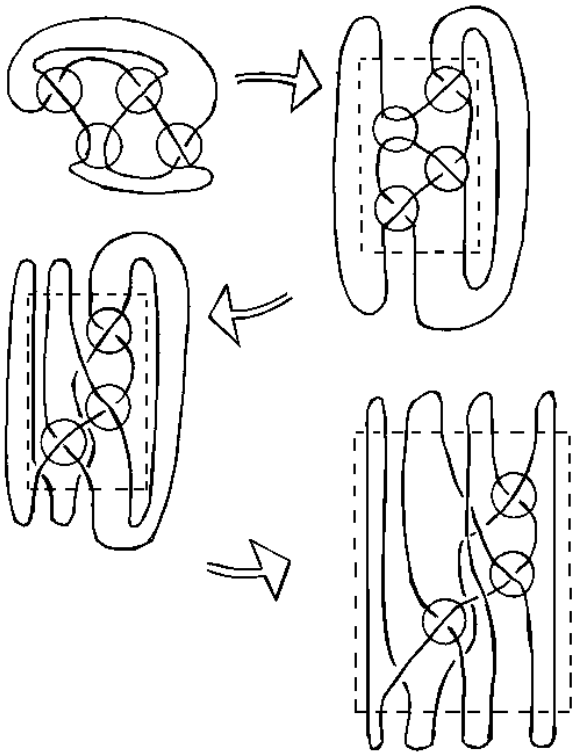


Figure 5.11: The conversion of a knot into a plait.

This algorithm may clearly be applied to our polyhedron $P(i, j)$. However we have the problem of the regions which depends upon the exact configuration of the knot. This can be solved by forcing the line L to intersect every region in the *polyhedron* and therefore intersecting some regions of the *knot* more than once. This is unfortunate but unavoidable if we are seeking a general solution of the problem. The manner in which this may be done most economically is illustrated in figure 5.12. The line L is the dotted line beginning at point A and finishing at point B. If the polyhedron has an odd number of columns (as the one in figure 5.12), then the line L is best described by the dotted line in figure 5.12. If however, the polyhedron has an even number of columns, then the line L is best described by the dotted line in figure 5.12 from point A to point C and then the dashed line from point C to point B. If algorithm 5.3.1 is correct then a line drawn in a general polyhedron $P(i, j)$ according to this example is a valid braiding axis.

We may find an axis which passes through every region exactly once, if possible, by the following algorithm.

Algorithm 5.3.2 *Input: A knot projection given in our notation. Output: An axis which passes through every region exactly once, if this is possible. If not the output is an axis which passes through each region at least once.*

1. Get the region information as prescribed in algorithm 5.2.3.

2. Construct a graph in which each region is symbolized by a node and two nodes are connected by an unweighted edge if they are adjacent in the plane.

3. A Hamiltonian circuit is then a path which passes through each region, that is node, exactly once starting in the infinite region and returning there. If a Hamiltonian circuit exists, so does an optimal axis. If no Hamiltonian circuit exists, we find an axis using algorithm 5.3.1 which gives an axis which passes through every region at least once.

The advantage is that we will generate a braid with less strings but the Hamiltonian circuit problem is NP-complete and so the execution of algorithm 5.3.2 is exponential (unless we use an

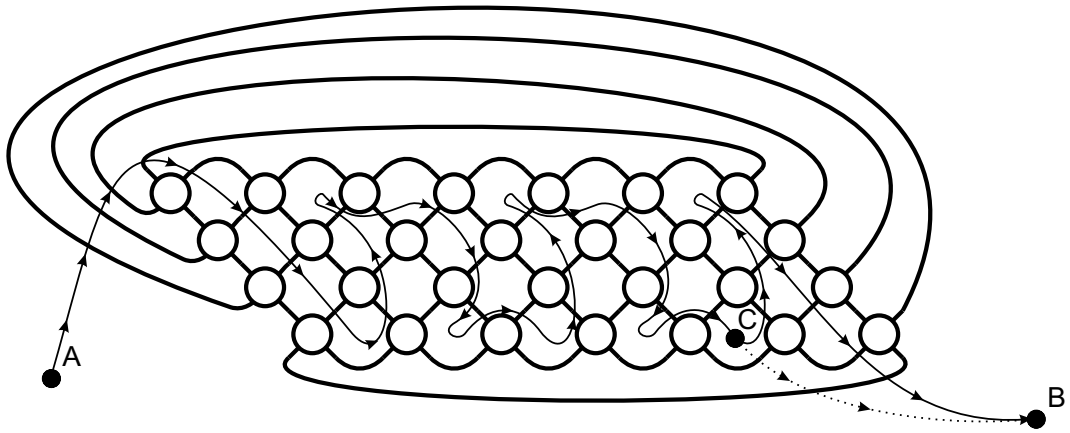


Figure 5.12: The axis of the braid through the polyhedron $P(i, j)$.

approximation algorithm or it is shown that P = NP). This fact lends further weight towards the usefulness of algorithm 5.3.1. The primary usefulness of this algorithm originates in the fact that the laying of the axis does not depend upon the exact knot configuration, only the labeling does. Before we continue, we prove that algorithm 5.3.1 always yields a valid axis, this essentially amounts to proving Alexander's theorem.

Theorem 5.3.3 *Given any knot projection, algorithm 5.3.1 will find an axis about which the knot is without local maxima or minima.*

Proof. Alexander's theorem [4] states given a knot projection, it is possible to deform it with respect to a point P in the projection plane that after the deformation a point A which travels along the knot in the direction of its orientation will travel around the axis defined by P (the axis is a line perpendicular to the projection plane intersecting it at P) in a constant fashion, either clockwise or counterclockwise, for the entire circumnavigation of the knot. We wish to do the opposite, namely to deform the axis around the knot projection to achieve the same ends. We can imagine the process of laying the axis as akin to sewing in which we move the needle up from and down onto the plane. Morton [115] has constructed a similar method to ours which he calls "threading."

The knot divides the plane into several regions. If the axis does not intersect a particular region, the point A will change course during traversing the knot and so the axis must intersect each region. It is however clearly only necessary for the axis to intersect the region once. Choose a line in the plane which intersects the axis. With respect to this line we can define an angular coordinate θ going around the axis. As point A must travel around the axis in a constant fashion it must, after it passes $\theta = 0$, reach $\theta = \pi$ before it once again reaches $\theta = 0$. This shows that the axis, in the projection plane, must over and undercross the knot alternately with respect to A. This fulfills the requirements of an axis and these are assured by algorithm 5.3.1 and thus the theorem is proven. □

5.3.4 Getting the Braid

Having obtained the axis, we must now simply put together all the pieces and construct the braid. This will be done via the following algorithm.

Algorithm 5.3.4 *Input: An axis L in a knot projection given in $P(i, j)$ using our notation. Output: A braid the canonical closure of which is ambient isotopic to the given knot.*

1. Consider an empty polyhedron $P(i, j)$ and label each edge by the row and column index of the vertex out of which it is emerging on the right side giving it the further label a if it is the top edge and b if it is the bottom edge. That is the top right hand edge coming out of the vertex $(1, 1)$ would be $(1, 1)_a$.

2. All edges which intersect the axis L at a positive crossing are to be numbered in order starting at point A; suppose there are k of these.

3. Starting at the numbered edges, use the traversal algorithm to follow each edge around the knot until another positive crossing with the axis L. All edges encountered are to be labeled with the same number as the original edge.

4. When all edges are numbered, we have identified the individual strings of the braid and numbered them in order. Assign a distance value of 1 to each edge in the polyhedron.

5. Traverse the knot again as in step 3 but this time stopping at each double point and extracting which labeled string passes over which other labeled string and at which distance value this occurs.

6. When the whole has been traversed, we have a list of crossings specifying which strings are involved, which string crosses over the other and at what distance from the bottom of the braid the crossing occurs. This information may be used readily to construct a colored braid, which may be converted easily into an Artin braid word.

7. We assess the string labels around the knot and calculate the permutation associated with the braid which winds around our axis. If this permutation is different from the permutation of the braid which we obtained in step 6, the residual permutation must be added to this braid in the form of extra crossings.

The number of crossings is increased in some circumstances by a small amount in step 7 of the algorithm. It is a fact that there exist knots of minimal crossing number n which have closed braid representatives all of which have crossing numbers greater than n [131]. Hence, step 7 is not a deficiency of the algorithm 5.3.4 but a fundamental necessity.

It is clear from Alexander's theorem[4] that this algorithm works. The number of strings used is the number of positive crossings of the axis with the knot which is equal to half the number of crossings. The number of crossings of the axis with the knot is

$$N_c = \begin{cases} 4i + (2i+2)\left\lfloor \frac{j-2}{2} \right\rfloor & j \text{ odd} \\ 2i + (2i+2)\left(\frac{j-2}{2}\right) + 2 & j \text{ even} \end{cases} \tag{5.16}$$

where $\lfloor x \rfloor$ is the greatest integer less than x. An analysis of the possibilities in oddness and evenness of i and j reveals that N_c is always even which is good since we must have an equal number of positive and negative crossings.

Algorithm 5.3.4 therefore finds a braid with a number of strings which scales linearly in the number of rows and columns necessary to represent the knot. It is conceivable that a more economical way of laying an axis may be found using algorithm 5.3.2 but this has an exponential complexity. The number of strings may be reduced after the braid has been found using Markov's theorem.

The determination of the regions, the laying of the axis, the labeling of the axis crossings, the labeling of the edges and the extraction of the double point information all take a time proportional to the number of vertices in the polyhedron ij. The building of the braid from the crossing information takes time proportional to ij. Therefore the entire algorithm to proceed from a knot projection to a canonically closed braid has complexity $O(ij)$. This algorithm succeeds in being readily implementable and in constructing a braid which is reasonably small.

5.4 Translation from Conway's Knotation

If the knot is given in Conway's notation [50], we may make the translation by fitting the appropriate basic polyhedron into $P(i, j)$ with a specific choice for i and j. Since Conway uses integral tangles for his notation, this method will yield a matrix with integer number entries. In the equations below we write Conway's notation and ours in correspondence, the letters imply integral tangles and the operator M is to be understood as the mirror operator from above. The equations may be verified readily by drawing the diagrams, they have been omitted here for reasons of space.

Using integral tangles in our matrix notation makes the notation more compact in that fewer rows and columns are needed to denote a knot but it also makes it more complex since each matrix element may take many values. We must have a method for separating out the integral tangles introduced into the notation via equations 5.17 - 5.25.

Algorithm 5.4.1 *Input: A matrix describing a knot in our notation in which one or more elements are integral tangles. Output: A matrix describing the same knot in which all elements are elementary tangles.*

1. Focus attention on the first integral tangle which is not elementary, suppose this has value sk where $s = \pm 1$ is the sign and k is a positive integer greater than one.
2. Create $k-1$ columns immediately after the column containing the current integral tangle and fill each new vertex with a 0 tangle.
3. Suppose the current integral tangle is $p_{mn} = sk$. Then set $p_{mq} = s$ for $n \leq q \leq n+k-1$.
4. Finally exchange the values of elements $p_{m+1\ n}$ and $p_{m+1\ n+k-1}$.

It is easy to convince oneself, by drawing a few diagrams, that algorithm 5.4.1 will accomplish the decomposition. While the number of rows stays constant, the number of columns may explode if there are numerous integral tangles of high values in the matrix. However this algorithm will give us a ready means, together with equations 5.17 - 5.25, to convert any knot given in Conway's notation into our notation.

$$1^*a = (a) \tag{5.17}$$

$$6^*a.b.c.d.e.f = \begin{pmatrix} a & c & e \\ M(b) & M(d) & M(f) \end{pmatrix} \tag{5.18}$$

$$6^{**}x.a.b.c.d.y = \begin{pmatrix} \infty & a & x \\ \infty & M(b) & \infty \\ \infty & c & \infty \\ M(y) & M(d) & \infty \end{pmatrix} \tag{5.19}$$

$$8^*a.b.c.d.e.f.g.h = \begin{pmatrix} a & c & e & g \\ M(b) & M(d) & M(f) & M(h) \end{pmatrix} \tag{5.20}$$

$$9^*a.b.c.d.e.f.g.h.i = \begin{pmatrix} a & d & g \\ M(b) & M(e) & M(h) \\ c & f & i \end{pmatrix} \tag{5.21}$$

$$10^*a.b.c.d.e.f.g.h.i.j = \begin{pmatrix} a & c & e & g & i \\ M(b) & M(d) & M(f) & M(h) & M(j) \end{pmatrix} \tag{5.22}$$

$$10^{**}a.b.c.d.e.f.g.h.i.j = \begin{pmatrix} a & c & e & 0 & 0 & \infty \\ M(b) & M(d) & M(f) & M(h) & M(j) & \infty \\ 0 & 0 & g & i & 0 & \infty \end{pmatrix} \tag{5.23}$$

$$10^{***}x.a.b.c.d.e.f.g.h.y = \begin{pmatrix} \infty & 0 & a & x \\ \infty & M(e) & M(b) & \infty \\ \infty & f & c & \infty \\ \infty & M(g) & M(d) & \infty \\ y & h & 0 & \infty \end{pmatrix} \tag{5.24}$$

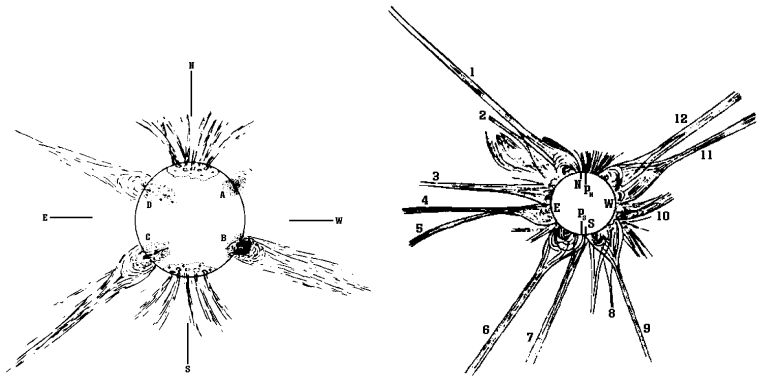
$$11^*a.b.c.d.e.f.g.h.i.j.k = \begin{pmatrix} a & c & e & g & i \\ M(b) & M(d) & M(f) & M(h) & M(j) \\ 0 & 0 & \infty & 0 & \infty \\ 0 & \infty & M(k) & 0 & \infty \end{pmatrix} \tag{5.25}$$

Chapter 6

The Solar Heating Problem

The sun is like an onion in that it has many layers. The core of the sun has a temperature of about 16 million degrees Celsius and the photosphere (the part that we actually see) only 6000 degrees. From this information, one would think that the temperature drops as one moves further from the core of the sun. This is false, however, as the corona, which is outside of the photosphere, has more than one million degrees. The temperature increases from a few thousand to a few million degrees in the space of about 500 kilometers. As no nuclear reactions take place in either the photosphere nor the corona, some other mechanism(s) must be responsible. Ever since the temperature of the corona was measured [68] in 1939, the solution to this problem is unknown. Many partial solutions are known, a plethora of models has been created and much effort spent. In this chapter we present a preliminary investigation of one mechanism.

6.1 Introduction



(a) (b)

Figure 6.1: Two schematic diagrams of the solar corona. Image (a) shows the evolution of a "typical" region in the corona through the stages: enhancement (A), streamer (B) to helmet streamer (C and D). Image (b) illustrates the structure of the corona on 19 June 1936 with various structures. It is shown clearly that some field lines are closed upon the solar surface while others appear to go to infinity. These images were taken from [119] and [35] respectively.

The corona is that layer of the sun in which the solar wind originates; it is the atmosphere of the sun and extends for about one million kilometers from the surface of the sun. We usually see the photosphere and can only see the corona during an eclipse when the moon covers the photosphere or with a specially built telescope. The huge temperatures in the corona require constant input of heat for without this input, the plasma would cool down in about one hour. It is a mystery from whence this heat comes. The corona's mass is an ionized plasma which acts like a fluid according

to many forces: gravity and heat convection, for example. For recent observational data about the corona, see for example [61] and [155]. Figure 6.1 is an illustration of the structure of the corona.

The most likely source for the heat is what has recently been called the magnetic carpet. The corona houses large magnetic fields in constant state of change. This carpet will be our focus. As we are dealing with a magnetic field inside a moving fluid like substance, we call this magnetohydrodynamics. A field of any kind may be mathematically treated in a variety of ways. We could consider it a distribution of vectors over space or we could draw lines through space which represent the integral curves of the field. It is these "field lines" that we shall consider. The topology of the field is defined as the topology of this collection of field lines. Strictly speaking, magnetic field lines may not have endpoints as this would give rise to a magnetic monopole (we shall not go into this heated discussion here but shall assume non-existence of magnetic monopoles until experimentally falsified). From observations, however, the lines appear to have endpoints as the lines go deep into the photosphere where we can no longer see them. Because we see only part of the field lines, we do not know how they connect underneath the surface of the photosphere. As we can not see the connections of the filaments beneath the photosphere, we are going to assume that they are not there. This is not to say that what we can not see can not affect the things we can see but we shall ignore them anyway. What we obtain therefore, are a collection of arches with footpoints on the photosphere. The surface on which the footpoints dance is covered by convection zones. These look like huge bubbles of the sort one gets when boiling pudding. The material is carried from the center of these zones to their boundary. Thus a footpoint is far more likely to be near or on a boundary of a zone than inside the zone itself. To make matters even worse, the zones rearrange themselves approximately every ten minutes.

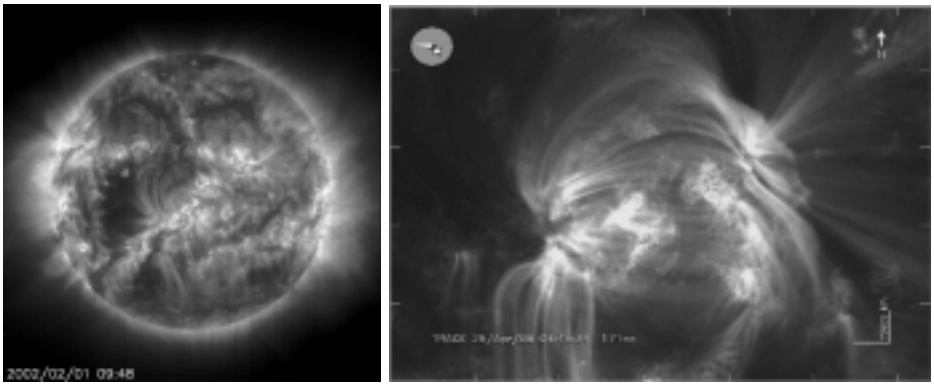


Figure 6.2: The left image shows the solar corona against the disk in one of the hydrogen line spectra and the right image shows a closeup of a collection of flux tubes where the field lines are clearly visible.

It is, of course, not obvious at all that we should be able to *see* field lines at all; they are, after all, a human intellectual construction. Indeed, we do not see the lines of the field as such but we see what are called filaments, which are thin and long local collections of plasma at a different temperature than the surroundings. It is because the temperature in the filament is cooler than in its surroundings that we can observe them and because of their thin and elongated structure. they look like lines (see figure 6.2). Conveniently, they are also parallel to the local magnetic field lines and so they provide a tracer for them. A good guide to what we can see and how to do it is given in [14].

The filaments are approximately parabolic and typically 70 Megameters long (in contrast to the sun's circumference of about 4400 Megameters). They are subject to gravity pulling the material towards the center of the sun. Pressure builds up driving the material toward the outside. Pressure $\rho(z)$ at distance z above the convection zone surface is given by

$$\rho(z) = \rho_0 e^{-gz/kT} \tag{6.1}$$

where ρ_0 is the pressure at the surface, $g = 1.92 \times 10^{13} ms^{-2}$ the acceleration due to gravity, $k = 1.38 \times 10^{-23} JK^{-1}$ Boltzmann's constant and T the temperature which we will set at one

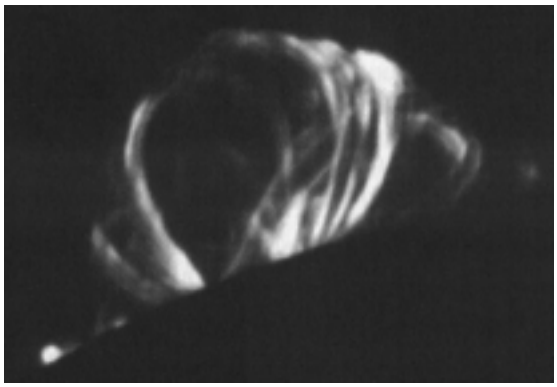


Figure 6.3: As an example of a solar loop prominence this photograph shows clearly where the field lines are and what the whole configuration looks like. From the dark arch at the bottom of the loop which is the solar surface, one can see how large the prominence is. Most observations are against the disk of the sun and then we can only see the arches from the top. This image was taken by Victor J. Lopez on 10 March 1981.

million degrees Kelvin. This defines an outwardly force which gets gradually weaker as the material is ejected.

The filaments themselves act as strings which relax elastically according to the models described in chapter 4. The filaments may also reconnect with one another and break into several pieces, effectively duplicating themselves. It is precisely this reconnection that is thought to generate the heat. During the evolution of a complex field topology a lot of energy is input and during reconnection most of that energy is rapidly dissipated into heat.

The filaments are thick tubes which contain hot plasma. The plasma moves along the axis of the filament from one footpoint to the other. The mass flow along the axis of the tube occurs at about 0.6 ms^{-1} near the footpoints on the photosphere [67] and between 1 and 2 ms^{-1} in regions 100 km higher [135]. Most models give rise to significant variation of the cross-sectional area of a filament (thickest at its highest point). This is however not observed. The observed filaments have thickness profiles which are sometimes thicker in the middle, sometimes thicker at the footpoints and sometimes of constant thickness, there are even filaments with a constantly increasing thickness from one footpoint to another [156]. The results of two distinct experiments (Yohkoh and TRACE) have given rise to these observations with the essential point that the difference between the thinnest and thickest point along a filament is, on average, only 15 percent of the thickness of the thinnest part [156] [94]. It is also observed that the density of the plasma inside a loop is constant over the cross-section [94].

The magnetic field inside a flux tube filament is about 0.2 Tesla (The solar physics community insists on using non-SI units, so that 1 Gauss = 0.0001 Tesla) which can be measured by a clever manipulation of the Zeeman effect [139]. The diameter of a flux tube has been measured to lie between 100 and 300 km [138]. All of these measurements come with large errors and are open to debate but they give us the order of magnitude region in which the parameter lie.

So called force-free magnetic fields are important in the corona [127] [163]. From the conception of force-free fields [101], invariants were found [157] and these lead to close investigation of the properties of these fields [42]. The structure of some such fields was investigated in [60] and it was found that they behave chaotically and are very unstable. This is particularly important as it is thought that force-free fields represent steady state minima of the magnetic field in the corona. A recent investigation of the force-free fields above the sun has lead to new solutions to the force-free field equations which could be promising [160]. In a simulation there are privileged field lines, namely those of zero field. This magnetic separator is important as it represents the borderline between regions of different connectivity. The topological evolution of these is considered in [37].

The filaments store and release energy, accelerate groups of electrons (heat the corona), eject mass (as flares and coronal mass ejections), emit radiation in the X-ray, EUV, UV, visible and radio bands and cause mass flow along the axis. These are the major features of a filament and

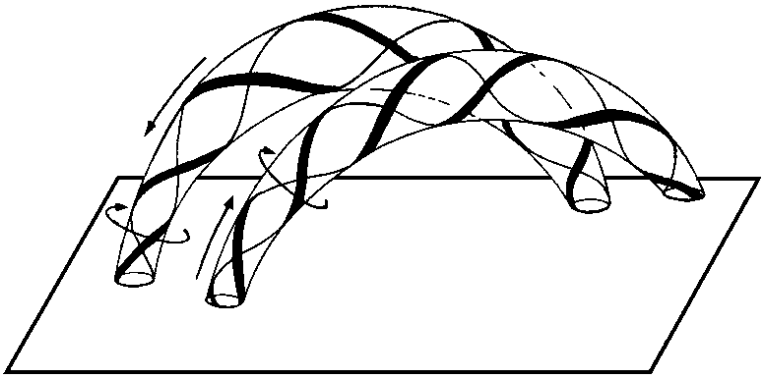


Figure 6.4: This schematic shows two filaments with similar axial twist but opposite flow direction. The diagram is taken from [140] and illustrates the Gold-Hoyle model [70] according to which these two filaments should reconnect thereby releasing energy and giving rise to a solar flare.

must be explained by a model as required and reviewed in [140]. Many of the physical processes of flux tube filaments and model are covered in [122] and a good review of the solar dynamo is given in [66]. The filaments differ in size and there are more small filaments than large ones and consequently there are more small flares than large ones. One might think that the sum total of the small events causes the bulk of the heating. This is a controversial point with evidence for both sides [1]. This suggests that present efforts into simulating and explaining the large events are well spent.

Reconnecting field lines can be indirectly observed through magnetograms and are thought to produce regions of canceling magnetic fields; reconnection can also be a mechanism for the observed mass flow along filaments [100]. A good review on the process of reconnection is given in [128].

This creates the background for the problem which we shall attempt to simulate. As this is a preliminary study, we shall consider the effects of the simulation on a particular configuration of arches, namely those of figure 6.2.

6.2 The Temperate Photosphere

One wonders though why the photosphere is so cool when it is next to a blazing furnace. What does it mean for a gas to have a high temperature? Temperature is a measure of the average kinetic energy of the gas atoms, that is, a measure of how fast they move. A high temperature gas has atoms with a larger average velocity than a low temperature gas of the same composition. We thus infer that the atoms in the corona are moving much more rapidly than those in the photosphere.

For the corona to heat the photosphere, the coronal gas must cause the photospheric atoms to move faster. It could do so by colliding and mixing with the cooler gas and thus transferring some of its kinetic energy. At a temperature of a million degrees, the gas in the corona is highly ionized, which means that neutral atoms no longer exist but rather freely moving electrons and atomic nuclei. Because electrons are much lighter than protons the hot electrons have very high speeds and could travel into the photospheric gas to collide with the atoms there, increasing their velocities. These two heating mechanisms are called convection and conduction, respectively. A gas at a few million degrees radiates energy; much of it is emitted in the form of very high-energy x-ray photons. X-ray photons impinging on the photosphere could also transfer energy to the gas atoms there. This heating mechanism is radiation.

Yet the three traditional methods of heating do not raise the photospheric temperature for a simple reason. Suppose we had a thermometer that could measure temperatures of millions of degrees and put it in the corona. In order to make a temperature measurement, the thermometer must be heated up by the coronal atoms, electrons or x-ray photons impinging upon it. The corona, however, has such a low density that the thermometer will almost never be hit. So while the thermometer is technically sitting in a gas that is at a few million degrees it does not know it. There are just not enough atoms to heat our hypothetical thermometer or the underlying

photosphere.

6.3 Generating Arches from Footpoints

For ease of use, the simulation will take a set of two dimensional coordinates as the locations for the footpoints and will generate the arches automatically. Clearly every arch has two footpoints. Consider a plane which intersects the two footpoints of a particular arch and is perpendicular to the surface; this specifies the plane exactly. The arch is to be drawn in this plane. We may choose, for simplicity, to take the distance between the two footpoints and to generate a semicircle with that diameter. Alternatively and more realistically, we may generate a parabola. To maintain maximum control over how high the arches are in relation to each other, we generate the parabolas from the semicircles by scaling such that the highest point of the semicircle is pulled upwards until it reaches some given parameter; the rest of the arch follows this scaling. This allows us to set certain profiles or to set the relative heights manually.

It is observed that the footpoints of most flux tube networks are very close to neutral lines in the magnetic field. For unknown reasons, these neutral lines look very much like integral signs and could thus be easily simulated using a Bezier line with two control points. We could thus envision generating footpoints automatically upon input of control points for the shape of a neutral line. This would require more input from observations however to discover just what the positional relationship between footpoints and neutral line is.

6.4 The Simulation

Having got the arches, we simulate them. The forces we shall consider for the moment are the pressure force, elastic force and repulsive force. The pressure force, as given in equation 6.1, is a height dependent force which tends to increase the size of a filament. The elastic and repulsive forces, as described in chapter 4, tend to reduce the size of a filament while preventing topology changes. The forces have to be balanced such that a filament will eventually reach a steady state. Clearly if one force dominated, the filament would continue to grow or shrink and this is not what is observed. When reconnection is to be simulated, the repulsive force would have to be switched off.

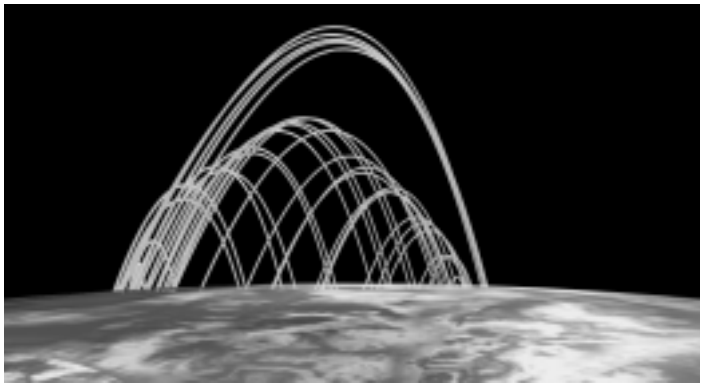


Figure 6.5: This figure displays the filaments which we created on the computer using footpoint data derived from observations. The line data was generated using the author's BraidLink and the raytracing was done using PovRay.

We begin with an observation (for example figure 6.2). From this we extract the footpoints and draw the arches to obtain a computer model of that situation, see figure 6.5. The observation which lead to the model in figure 6.2 was made in [106] and the model was originally realized using wire.

6.5 Mutual and Self Helicity

A field line has a positive and a negative footpoint. If $\hat{\mathbf{n}}$ is a unit normal to the surface of the sun S, then a positive footpoint is a footpoint for which $\mathbf{B} \cdot \hat{\mathbf{n}}|_S > 0$ and negative if the inequality is reversed. We have a null point when $\mathbf{B} \cdot \hat{\mathbf{n}}|_S = 0$. For a pair of flux tubes, we can have the two situations displayed in figure 6.6. If the tubes do not cross, then knowledge of the footpoint coordinates is sufficient; if they do, then we also need to know which tube overcrosses the other. The mutual helicity between two tubes is given by [15]

$$H_{ij} = H_{ji} = \frac{\Phi_i \Phi_j}{2\pi}(\alpha + \beta) \tag{6.2}$$

where Φ_k is the magnetic flux in tube k and the angles α and β are defined in figure 6.6 (note that the top tube is labeled 1 in figure 6.6).

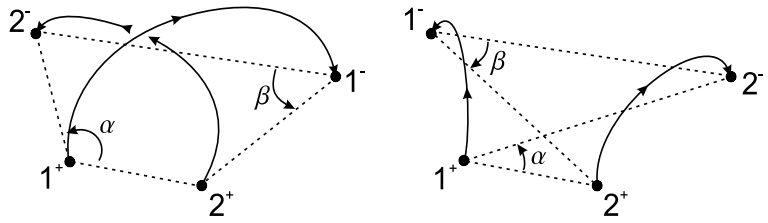


Figure 6.6: The angles which must be known to compute the mutual helicity (see equation 6.2) of two flux tubes are defined here for the two situations possible: crossing tubes and non-crossing tubes. If the tubes cross, we assume that the top one is labeled 1.

It is also possible for a flux tube to be twisted in itself. This can be clearly seen in figure 6.7. This internal twist is called self-helicity [158]. If a field line encircles the axis of the i^{th} tube T_i times, then we define its self-helicity as

$$H_{ii} = T_i \Phi_i \tag{6.3}$$

A collection of N flux tubes then has a total helicity which is the sum of the pairwise mutual helicities and the individual self-helicities,

$$H = \sum_{i=1}^{N} H_{ii} + \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} H_{ij} \tag{6.4}$$

The matrix with entries H_{ij} is symmetric about the diagonal and can be conviniently used to summarize the helicities of a large flux tube network as the sum of its entries is the total helicity of the network. So if a super-network is made up of networks, we can represent the helicity of the super-network in terms of the helicities of the networks in such a matrix. Consider our current situation in figure 6.5. The flux tubes can be loosely divided into three groups: small, medium and large tubes. If we label each group by A, B and C respectively, then the helicity matrix of the whole super-network is

$$H = \begin{pmatrix} H_{AA} & H_{AB} & H_{AC} \\ H_{BA} & H_{BB} & H_{BC} \\ H_{CA} & H_{CB} & H_{CC} \end{pmatrix} \tag{6.5}$$

Observationally, it is difficult to decide which flux tube overcrosses the other and also a measurement of flux is not accurate. For our model, we shall assume that all tubes have identical flux. If a pair of flux tubes has mutual helicity H_{ij} and we were to switch which tube is on top of the other without changing the position of the footpoints, then the mutual helicity would become $H_{ij} - \Phi_i \Phi_j$. The reason is that the tube labeled 1 in figure 6.6 is the top tube. If this switches, the angles α and β go to the two unnamed angles in the figure, α' and β', say. Clearly, we have

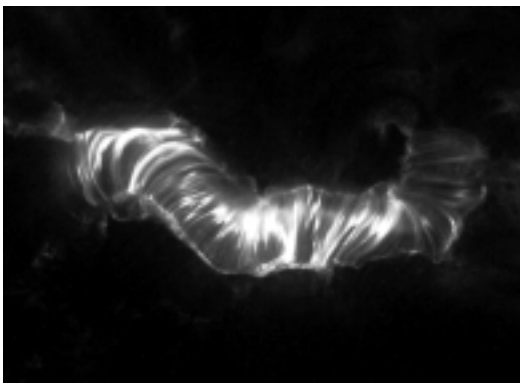


Figure 6.7: This filament shows clearly the presence of axial twist. (This TRACE image of AR9077 shortly after the flare erupted is available as Astronomy Picture of the Day from http://antwrp.gsfc.nasa.gov/apod/ap000720.html.)

$\alpha + \beta - (\alpha' + \beta') = 2\pi$ and so

$$H_{ij} = \frac{\Phi_i \Phi_j}{2\pi}(\alpha + \beta) \longrightarrow H'_{ij} = \frac{\Phi_i \Phi_j}{2\pi}(\alpha' + \beta') \tag{6.6}$$

$$= \frac{\Phi_i \Phi_j}{2\pi}(\alpha + \beta - 2\pi) \tag{6.7}$$

$$= H_{ij} - \Phi_i \Phi_j \tag{6.8}$$

The most important point to note about helicity is that it is conserved throughout the evolution of field lines. In observational terms, the change of helicity in the sun occurs at such long timescales compared to the lifetime of flux tubes that it is, for all practical purposes, conserved indeed.

6.6 The Moving Footpoints

Moving footpoints are thought to be a driving force of magnetohydrodynamic turbulence for the filaments. Some recent models and results are presented in [72] and references therein. The simplest way to move the footpoints is to choose an unbiased random walk. The footpoints move on the two dimensional surface of the photosphere which we regard as a plane. We choose four random numbers per tube within the range $[-\delta, \delta]$ where δ is a parameter of the model and add these numbers to the two coordinates of the footpoints to generate new footpoints. According to a well known result by Polya, such a random walk in one or two dimensions will eventually return to the original point; the probability for that to happen in three dimensions is roughly one-third.

An unbiased random walk is not really realistic as the footpoints are usually forced to approach and then remain on the boundary of the convection granules and as such we require a more complex random walk. Such random walks are discussed in [41] and will be used in the future to study this model.

6.7 Conclusions

We have simulated a model of filaments derived observational footpoint data. Our model was very simplistic but a steady state was achieved and helicity conserved throughout. The balancing of the forces essentially defines a plane $z = a$ for some constant a which depends on the surface pressure ρ_0 and the strength of the elastic force to which the highest points of the filaments tend. The simulation is encouraging but more realistic elements are needed.

Firstly the footpoint motion must become more like footpoint motion on a restricted domain with a preferred direction. The other environmental forces must be taken into account. There are many of these but the most crucial is the background magnetic field in which the lines sit which could be simulated by stationary field lines. There must also be more interaction between the field

lines as the only interaction at present is the repulsive force which is purely artificial in any case. Reconnection should also be taken into account as should the division of tubes into several tubes. Computational provision for some of these features has been made already. A more substantial collaboration with observational astrophysicists is needed in the future to make the simulation realistic and thus useful.

Bibliography

[1] Aschwanden, M. J. 1999 *Do EUV Nanoflares account for Coronal Heating?* Solar Physics **190**, 233 - 247. Cited on: 85

[2] Adian, S. I. 1957 *The unsolvability of certain algorithmic problems in the theory of groups* Trudy Moskov. Mat. Obsc. **6**, 231 - 298. Cited on: 14, 28

[3] Adian, S. I. 1957 *Finitely Presented Group and Algorithms* Dokl. Akad. Nauk SSSR **117**, 9 - 12. Cited on: 14, 28

[4] Alexander, J. W. 1923 *A lemma on systems of knotted curves* Proc. Nat. Acad. Sci. USA **9**, 93 - 95. Cited on: 17, 79, 80

[5] Appel, K. I. and Schupp, P. E. 1972 *The Conjugacy Problem for the Group of any Tame Alternating Knot is Solvable* Proc. Amer. Math. Soc. **33**, 329 - 336. Cited on: 14

[6] Arteca, G. A. 1999 *Path-integral calculation of the mean number of overcrossings in an entangled polymer network* J. Chemical Information & Computer Sci. **39**, 550 - 557. Cited on: 52

[7] Artin, E. 1925 *Theorie der Zöpfe* (in German) Abh. Math. Sem. Univ. Hamburg **4**, 47 - 72. Cited on: 17

[8] Artin, E. 1947 *Theory of braids* Ann. Math. **48**, 101 - 126. Cited on: 17, 19

[9] Baader, F. and Nipkow, T. 1998 *Term Rewriting and All That* (Cambridge University Press, Cambridge). Cited on: 27, 28, 29, 32, 35

[10] Bangert, P. D., Berger, M. A. and Prandi, R. 2002 *In Search of Minimal Random Braid Configurations* J. Phys. A: Math. Gen. **35**, 43 - 59. Cited on: 10

[11] Barnes, C. W. and Sturrock, P. A. 1972 *Force-free Magnetic-field Structures and their Role in Solar Activity* Astrophys. J. **174**, 659 - 670. Cited on: 58

[12] Bauer, G. and Otto, F. 1984 *Finite Complete Rewriting Systems and the Complexity of the Word Problem* Acta Inf. **21**, 521 - 540. Cited on: 28

[13] Baumslag, G., Boone, W. W. and Neuman, B. H. 1959 *Some Unsolvable Problems about Elements and Subgroups of Groups* Math. Scand. **7**, 191 - 201. Cited on: 28

[14] Beck, R., Hilbrecht, H., Reinsch, K. and Völker, P. 1995 *Solar Astronomy Handbook* (William-Bell Inc., Richmond, VA, USA). Cited on: 83

[15] Berger, M. A. 1984 Geophys. and Astrophys. Fluid. Dyn. **30**, 79. Cited on: 87

[16] Berger, M. A. 1993 *Energy-crossing number relations for braided magnetic fields* Phys. Rev. Lett. **70**, 705 - 708. Cited on: 52, 53, 56

[17] Berger, M. A. 1994 *Minimum crossing numbers for 3-braids* J. Phys. A **27**, 6205 - 6213. Cited on: 41, 50, 52, 55, 60

[18] Berger, M. A. 1994 *Coronal Heating by Dissipation of Magnetic Structure* Space Science Reviews **68**, 3. Cited on:

[19] Birkhoff, G. 1935 *On the structure of abstract algebras* Proc. Camb. Phil. Soc. **31**, 433 - 454. Cited on: 28

[20] Birman, J. S. 1974 *Braids, Links and Mapping Class Groups* Ann. of Math. Studies 82 (Princeton Univ. Press, Princeton). Cited on: 18, 19, 20

[21] Birman, J. S. and Hirsch, M. D. 1998 *A New Algorithm for Recognizing the Unknot* Geom. and Topol. **2**, 178 - 220. Cited on: 13

[22] Birman, J. S., Ko, K. H. and Lee, S. J. 1998 *A New Approach to the Word and Conjugacy Problems in the Braid Groups* Ad. Math. **139**, 322 - 353. Cited on: 19, 40, 47, 53, 54, 75

[23] Birman, J. S. and Menasco, W. 1992 *Studying Links Via Closed Braids I: A Finiteness Theorem* Pacific J. Math. **154**, 17 - 36. Cited on: 21

[24] Birman, J. S. and Menasco, W. 1991 *Studying Links Via Closed Braids II: On a Theorem of Bennequin* Topology and Its Applications **40**, 71 - 82. Cited on: 21

[25] Birman, J. S. and Menasco, W. 1993 *Studying Links Via Closed Braids III: Classifying Links which are Closed 3-braids* Pacific J. Math. **161**, 25 - 113. Cited on: 21

[26] Birman, J. S. and Menasco, W. 1990 *Studying Links Via Closed Braids IV: Closed Braid Representatives of Split and Composite Links* Invent. Math. **102**, 115 - 139. Cited on: 21

[27] Birman, J. S. and Menasco, W. 1992 *Studying Links Via Closed Braids V: Closed Braid Representatives of the Unlink* Trans. AMS **329**, 585 - 606. Cited on: 21

[28] Birman, J. S. and Menasco, W. 1992 *Studying Links Via Closed Braids VI: A Non-Finiteness Theorem* Pacific J. Math. **156**, 265 - 285. Cited on: 21

[29] Birman, J. S. and Menasco, W. 1992 *A calculus on links in the 3-sphere* in Kawauchi, A. *Knots 90* (Walter de Gruyter, Berlin), 625 - 631. Cited on: 21

[30] Birman, J. S. and Wajnryb, B. 1986 *Markov Classes in Certain Finite Quotients of Artin's Braid Group* Israel J. Math. **56**, 160 - 178. Cited on: 21

[31] Book, R. V. 1987 *Thue Systems as Rewriting Systems* J. Sym. Comp. **3**, 39 - 68. Reprinted in *Rewriting Techniques and Applications* ed. by Jouannaud, J.-P. (Academic Press, London), 39 - 68. Cited on: 28

[32] Boone, W. W. 1954 *Certain Simple Unsolvable Problems in Group Theory I, II* Nederl. Akad. Wetensch. Proc. Ser. A **57**, 231 - 237, 492 - 497. 1955 *Certain Simple Unsolvable Problems in Group Theory III, IV* **58**, 252 - 256, 571 - 577. 1957 *Certain Simple Unsolvable Problems in Group Theory V, VI* **60**, 22 - 27, 227 - 232. Cited on: 28

[33] Boone, W. W. and Rogers, H. Jr. 1966 *On a Problem of J. H. C. Whitehead and a Problem of Alonzo Church* Math. Scand. **19**, 185 - 192. Cited on: 28

[34] Boyland, P. L., Aref, H. and Stremler, M. A. 2000 *Topological fluid mechanics of stirring* J. Fluid Mech. **403**, 277 - 304. Cited on: 52

[35] Bronshten, V. A. 1959 *The Structure of the Far Outer Corona of June 19, 1936* Astro. J. **36**, 845 - 850. Cited on: 82

[36] Brown, M. R., Canfield, R. C. and Pevtsov, A. A. ed. by 1999 *Magnetic Helicity in Space and Laboratory Plasmas*, AGU Geophysical Monograph Series 111 (American Geophysical Union, Washington). Cited on: 52

[37] Brown, D. S. and Priest, E. R. 1999 *The Topological Behaviour of Stable Magnetic Separators* Solar Physics **190**, 25 - 33. Cited on: 84

[38] Buchberger, B. 1987 *History and Basic Features of the Critical-Pair/Completion Procedure* J. Sym. Comp. **3**, 3 - 38. Printed in *Rewriting Techniques and Applications* ed. by Jouannaud, J.-P. (Academic Press, London), 3 - 38. Cited on: 29, 30

[39] Burde, G., Zieschang, H. 1985 *Knots* (Walter de Gruyter, Berlin). Cited on: 13, 14, 15, 17, 69

[40] Chan, T. 2000 *HOMFLY polynomials of some generalized Hopf links* J. Knot Th. Rami. **9**, 865 - 883. Cited on: 25

[41] Chandrasekhar, S. 1943 *Stochastic problems in Physics and Astronomy* Reviews of Modern Physics **15**, 1. Reprinted in Wax, N. 1954 *Noise and stochastic processes* (Dover, New York). Cited on: 88

[42] Chandrasekhar, S. and Woltjer, L. 1958 *On Force-Free Magnetic Fields* Proc. Natl. Acad. Sci. **44**, 285 - 289. Cited on: 84

[43] Chow, W. L. 1948 *On the algebraic braid group*, An. Math. **49**, 654 - 658. Cited on: 18, 19

[44] Chui, A. Y. K. and Moffatt, H. K. 1995 *The energy and helicity of knotted magnetic flux tubes* Proc. R. Soc. London A **451**, 609 - 629. Cited on: 53

[45] Clapham, C. R. J. 1967 *An Embedding Theorem for Finitely Generated Groups* Proc. Lond. Math. Soc. **3**, 419 - 430. Cited on: 28

[46] Clausen, S., Helgesen, G. and Skjeltorp, A. T. 1998 *Braid description of collective fluctuations in a few-body system* Physical Review E **58**, 4229 - 4237. Cited on: 52

[47] Cohen, D. E. 1987 *Computability and Logic* (Ellis Horwood, Chichester). Cited on: 28

[48] Collins, D. J. 1972 *Representation of Turing reducibility by word and conjugacy problems in finitely presented groups* Act. Math. **128**, 73 - 90. Cited on: 33

[49] Collins, D. J. 1980 *Conjugacy and the Higman Embedding Theorem* in *Word Problems II* Vol. 95 of Studies in Logic and the Foundations of Mathematics ed. by Adian, S. I., Boone, W. W., Higman, G. (North Holland, Amsterdam), 81 - 85. Cited on: 28

[50] Conway, J. H. 1970 *An Enumeration of Knot and Links, and Some of their Algebraic Properties* in Leech, J. 1970 *Computational Problems in Abstract Algebra* (Pergamon, Oxford). Cited on: 16, 69, 80

[51] Coxeter, H. S. M. and Moser, W. O. J. 1957 *Generators and Relations for Discrete Groups* (Springer, Berlin). Cited on: 19, 26, 28, 34

[52] Dai, X. and Diao, Y. 2000 *The Minimum of Knot Energy Functions* J. Knot Th. Rami. **9**, 713 - 724. Cited on: 53

[53] Dehn, M. 1911 *Über unendliche diskontinuerliche Gruppen* Math. Ann. **69**, 116 - 144. Cited on: 28

[54] Dershowitz, N. 1979 *A note on simplification orderings* Inform. Proc. Let. **9**, 212 - 215. Cited on: 29

[55] Dershowitz, N. 1981 *Termination of linear rewriting systems* in *Automata, Languages and Programming* ed. by Even, S. and Kariv, O., Lecture Notes in Computer Science Volume 115 (Springer, Heidelberg), 448 - 458. Cited on: 29

[56] Dershowitz, N. and Jouannaud, J.-P. 1990 *Rewrite Systems* in *Handbook of Theoretical Computer Science Volume B: Formal Methods and Semantics* ed. by van Leeuwen, J. (North-Holland, Amsterdam), 243 - 320. Cited on: 27

[57] Dershowitz, N. 1987 *Termination of Rewriting* J. Sym. Comp. **3**, 69 - 116. Printed in *Rewriting Techniques and Applications* ed. by Jouannaud, J.-P. (Academic Press, London), 69 - 116. Cited on: 29

[58] Dowker, C. H. and Thistlethwaite, M. B. 1983 *Classification of Knot Projections* Topology and Its Applications **16**, 19 - 31. Cited on: 13, 16

[59] Epstein, D. B. A., Cannon, J. W., Holt, D. F., Levy, S. V. F., Paterson, M. S., Thurston, W. P. 1992 *Word Processing in Groups* (Jones and Bartlett, Boston). Cited on: 19, 41

[60] Evangelidis, E. A., Vaughan, L. L. and Botha, G. J. J. 2000 *The Structure of Force-Free Magnetic Fields* Solar Physics **193**, 17 - 32. Cited on: 84

[61] Foley, C. A. de R. S. 1998 *The Temperature Structure of the Quiescent Regions of hte Solar Corona* unpublished Ph.D. Thesis, Mullard Space Science Laboratory, University College London. Cited on: 83

[62] Freedman, M. H. and He, Z. X. 1991 *Divergence-free fields – energy and asymptotic crossing number* Ann. Math. **134**, 189 - 229. Cited on: 53

[63] Garey, M. R. and Johnson, D. S. 1979 *Computers and Intractability* (W. H. Freeman, New York). Cited on: 28, 42, 44, 45

[64] Garside, F. A. 1969 *The braid group and other groups* Quart. J. Math. Oxford **20**, 235 - 254. Cited on: 17, 18, 19, 21, 22, 48

[65] Gilbert, N. D. and Porter, T. 1994 *Knots and Surfaces* (Oxford University Press, Oxford). Cited on: 15

[66] Gilman, P. A. 1986 *The Solar Dynamo: Observations and Theories* in Sturrock, P. A. (ed. by) 1986 *Physics of the Sun, Volume I: The Solar Interior* (D. Reidel Pub. Co., Dordrecht, Holland). Cited on: 85

[67] Giovanelli, R. G. and Slaughter, C. 1978 *Motions in Solar Magnetic Tubes I. The Downflow* Solar Physics **57**, 255 - 260. Cited on: 84

[68] Grotrian, W. 1939 Naturwissenschaften **27** 214. Cited on: 82

[69] Gordon, C. personal communication. Cited on: 13

[70] Gold, T. and Hoyle, F. 1960 Monthly Notices Roy. Astron. Soc. **120**, 89. Cited on: 85

[71] Goldman, J. R. and Kauffman, L. H. 1997 *Rational Tangles* Adv. Appl. Math. **18**, 300 - 332. Cited on: 69

[72] Gomez, D. O., Dmitruk, P. A. and Milano, L. J. 2000 *Recent Theoretical Results on Coronal Heating* Solar Physics **195**, 347 - 366. Cited on: 88

[73] Gutin, G. and Yeo, A. *Polynomial approximation algorithms for the TSP and the QAP with a factorial domination number* to appear in Discrete Applied Mathematics. Cited on: 43

[74] Haken, W. 1961 *Theorie der Normalflächen* Acta Math. **105**, 245 - 375. Cited on: 13

[75] Haken, W. 1962 *Über das Homöopathieproblem der 3-Mannigfaltigkeiten* Math. Z. **80**, 89 - 120. Cited on: 13

[76] Hass, J. 1998 *Algorithms for recognizing knots and 3-manifolds* Chaos Solitons Fractals **9**, 569 - 581. Cited on:

[77] Hemion, G. 1992 *The Classification of Knots and 3-dimensional Spaces* (Oxford Univ. Press, Oxford). Cited on: 13, 14

[78] Hempel, J. 1976 *3-manifolds* Ann. of Math. Studies Volume 86 (Princeton Uni. Press, Princeton). Cited on: 14

[79] Higman, G. 1961 *Subgroups of finitely presented groups* Proc. Roy. Soc. London Ser. A **262**, 455 - 475. Cited on: 28

[80] Huet, G. 1980 *Confluent reductions: Abstract properties and applications to term rewriting systems* J. Assoc. Comput. Mach. **27**, 797 - 821. Cited on: 29, 36

[81] Huet, G. 1981 *A Complete Proof of the Knuth-Bendix Completion Algorithm* J. Comp. Syst. Sci. **23**, 11 - 21. Cited on: 30

[82] Huet, G. and Lankford, D. S. 1978, *On the uniform halting problem for term rewriting systems*, Rapport laboria 283, Institut de Recherche en Informatique et en Automatique, Le Chesnay, France. Cited on: 29

[83] Jacquemard, A. 1990 *About the effective classification of conjugacy classes of braids* J. Pure Appl. Al. **63**, 161 - 169. Cited on: 19

[84] Jantzen, M. 1997 *Basics of Term Rewriting*, in *Handbook of Formal Languages Volume 3: Beyond Words* ed. by Rozenberg, G. and Salomaa, A. (Springer, Heidelberg), 269 - 338. Cited on: 27

[85] Ji, H. T. 1999 *Turbulent dynamos and magnetic helicity* Phys. Rev. Lett. *83*, 3198. Cited on:

[86] Johnson, D. L. 1980 *Topics in the Theory of Group Presentations* Lon. Math. Soc. Lec. Notes Vol. 42 (Cambridge Uni. Press, Cambridge). Cited on: 33

[87] Jouannoud, J.-P. and Kirchner, H. 1986 *Completion of a set of rules modulo a set of equations* SIAM. J. Comp. **15**, 1155 - 1194. Cited on: 29

[88] Juhasz, A. 1992 *Solution of the Conjugacy Problem in One Relator Groups* in *Algorithms and Classification in Combinatorial Group Theory* ed. by Baumslag, G. and Miller, C. F. III, Math. Sci. Re. Inst. Pub. Volume 23 (Springer, New York), 69 - 81. Cited on: 28

[89] Kang, E. S. *et. al.* 1997 *Band-generator presentation for the 4-braid group* Topology and Its Applications **78**, 39 - 60. Cited on: 41

[90] Katritch, V., Bednar, J., Michoud, D., Scharein, R. G., Dubochet, J., Stasiak, A. 1996 *Geometry and physics of knots* Nature **384**, 142 - 145. Cited on: 52, 53

[91] Kauffman, L. H. 1987 *On Knots* Ann. Math. Studies Volume 115 (Princeton Uni. Press, Princeton). Cited on: 13

[92] Kawauchi, A. 1996 *A Survey of Knot Theory* (Birkhäuser Verlag, Basel). Cited on: 15

[93] Kirkman, Rev. T. P. 1884 *The Enumeration, Description, and Construction of Knots of Fewer than Ten Crossings* Trans. Roy. Soc. Edinburgh **32**, 281 - 309. Cited on: 11, 12

[94] Klimchuk, J. A. 2000 *Cross-sectional Properties of Coronal Loops* Solar Physics **193**, 53 - 75. Cited on: 84

[95] Klop, J. W. 1987 *Term rewriting systems: A tutorial*, Bulletin of the European Association for Theoretical Computer Science, **32**, 143 - 183. Cited on: 27

[96] Knuth, D. E. and Bendix, P. B. 1970 *Simple word problems in universal algebras* in *Computational Problems in Abstract Algebra* ed. by Leech, J. (Pergamon Press, Oxford), 263 - 297. Reprinted in 1983 in *Automation of Reasoning 2* (Springer, Berlin), 342 - 376. Cited on: 29, 30

[97] Lambropoulou, S. S. F. 1993 *A Study of Braid in 3-manifolds* unpublished PhD thesis (Univ. of Warwick). Cited on: 20, 75

[98] Lambropoulou, S. S. F. and Rourke, C. P. 1997 *Markov's Theorem in 3-manifolds* Topology and Its Applications **78**, 95 - 122. Cited on: 20

[99] Lickorish, W. B. R. 1988 *Polynomials for Links* Bull. London Math. Soc. **20**, 558 - 588. Cited on: 13

[100] Litvinenko, Y. E. and Martin, S. F. 1999 *Magnetic Reconnection as the Cause of a Photospheric Cancelling Feature and Mass Flows in a Filament* Solar Physics **190**, 45 - 58. Cited on: 85

[101] Lüst, R. and Schlüter, A. 1954 Z. Astrophys. **34**, 263. Cited on: 84

[102] Lyndon, C. R. and Schupp, P. E. 1977 *Combinatorial Group Theory* (Springer, Berlin). Cited on: 28

[103] Madlener, K. and Otto, F. 1985 *Pseudo-Natural Algorithms for Decision Problems in Certain Types of String Rewriting Systems* J. Sym. Comp. **1**, 383 - 418. Cited on: 28

[104] The CiME system is available from `http://cime.lri.fr`. Cited on: 30

[105] Markov, A. A. 1935 *Über die freie Äquivalenz geschlossener Zöpfe* (in German), Recueil Mathematique Moscou **1**, 73 - 78. [Mat. Sb. **43** 1936.] Cited on: 20

[106] Martin, S. F. and McAllister, 1997 in *Coronal Mass Ejections* ed. by Crooker, N., Josselyn, J.-A. and Feynman, J. (Geophys. Monog. 99, Amer. Geophys. Union) 127. Cited on: 86

[107] McCool, J. 1980 *On Reducible Braids* in *Word Problems II* ed. by Adian, S. I., Boone, W. W. and Higman, G. (North-Holland, Amsterdam), 261 - 295. Cited on: 21

[108] McRobie, F. A. and Thompson, J. M. T. 1993 *Braids and knots in driven oscillators* Int. J. Bifurcation Chaos **3**, 1343 - 1362. Cited on: 52

[109] Miller, C. F. III 1971 *On Group-theoretic Decision Problems and their Classification* Ann. Math. Stud. Volume 68 (Princeton Uni. Press, Princeton) Cited on: 28

[110] Miller, C. F. III 1992 *Decision Problems for Groups - Survey and Reflections* in *Algorithms and Classification in Combinatorial Group Theory* ed. by Baumslag, G. and Miller, C. F. III, Math. Sci. Re. Inst. Pub. Volume 23 (Springer, New York), 1 - 59. Cited on: 28, 34

[111] Moffatt, H. K. 1985 *Magnetostatic Equilibria and analogous Euler flows of arbitrarily complex topology. 1. Fundamentals* J. Fluid Mech. **150**, 359 - 378. Cited on: 53

[112] Moffatt, H. K. 1990 *The energy spectrum of knots and links* Nature **347**, 367 - 369. Cited on: 53

[113] Moore, C. 1993 *Braids in classical dynamics* Phys. Rev. Lett. **70**, 3675 - 3679. Stud. Logic Found. Math. Volume 95 (North-Holland, Amsterdam), 261 - 295. Cited on: 52

[114] Morton, H. R. 1983 *An irreducible 4-string braid with unknotted closure* Math. Proc. Camb. Phil. Soc. **93**, 259 - 261. Cited on: 20

[115] Morton, H. R. 1986 *Threading Knot Diagrams* Math. Proc. Camb. Phil. Soc. **99**, 247 - 260. Cited on: 79

[116] Murasugi, K. 1996 *Knot Theory and Its Applications* (Birkhäuser, Boston). Cited on: 13, 69

[117] Murasugi, K. and Kurpita, B. I. 1999 *A Study of Braids* Mathematics and Its Applications Series Volume 484 (Kluwer Academic Pub., Dordrecht). Cited on: 19

[118] Murasugi, K. and Thomas, R. S. D. 1972 *Isotopic closed nonconjugate braids* Proc. Am. Math. Soc. **33**, 137 - 139. Cited on: 20

[119] Newkirk, E. 1967 *Structure of the Solar Corona* Ann. Rev. Astron. Astrophys. **5**, 213. Cited on: 82

[120] Newman, M. H. A. 1942 *On theories with a combinatorial definition of 'equivalence'* Ann. Math. **43**, 223 - 243. Cited on: 29, 36

[121] Novikov, P. S. 1955 *On the Algorithmic Unsolvability of the Word Problem in Group Theory* Trudy Math. Inst. Steklov **44**, 1 - 143. Cited on: 28

[122] Parker, E. N. 1979 *Cosmical Magnetic Fields* (Oxford University Press, New York). Cited on: 85

[123] Parker, E. N. 1983 *Magnetic neutral sheets in evolving fields. II - Formation of the solar corona* Astroph. J. **264**, 642 - 647. Cited on: 52

[124] Parnell, C. E. and Jupp, P. E. 2000 *Statistical analysis of the energy distribution of nanoflares in the quiet Sun* Astroph. J. **529**, 554 - 569. Cited on: 52

[125] Paterson, M. S. and Razborov, A. A. 1991 *The set of minimal braids is co-NP-complete* J. Algorithms **12**, 393 - 408. Cited on: 41, 44, 45, 46, 47, 60, 65

[126] Pedersen, J. and Yoder, M. 1994 *Term Rewriting for the Conjugacy Problem and the Braid Groups* J. Symbolic Computation **18**, 563 - 572. Cited on: 38

[127] Priest, E. R. 1984 *Solar Magnetohydrodynamics* (D. Reidel Pub. Co., Dordrecht, Holland). Cited on: 84

[128] Priest, E. R. and Schrijver, C. J. 1999 *Aspects of Three-Dimensional Magnetic Reconnection* Solar Physics **190**, 1 - 24. Cited on: 85

[129] Rabin, M. O. 1958 *Recursive Unsolvability of Group Theoretic Problems* Ann. Math. **67**, 172 - 194. Cited on: 14, 28

[130] Reidemeister, K. 1926 *Elementare Begründung der Knoten-theorie* (in German), Abh. Math. Sem. Univ. Hamburg **5**, 24 - 32. Cited on: 13

[131] Ricca, R. L. 1998 *Applications of Knot Theory in Fluid Mechanics* in Jones, V. F. R. *et. al.*, ed. by *Knot Theory* Banach Center Pub. Vol. 42 (Inst. of Math., Polish Acad. Sci., Warszawa), 321 - 346. Cited on: 75, 76, 80

[132] Schubert, H. 1948 *Die Eindeutige Zerlegbarkeit eines Knotens in Primknoten* (in German), Sitz. Heidelberger Akad. Wiss., math.-nat. Kl., 55 - 104. Cited on: 21

[133] Schubert, H. 1961 *Bestimmung der Primfaktorzerlegung von Verkettungen* (in German), Math. Zeitschr. **76**, 116 - 148. Cited on: 21

[134] Shimamura, M. K. and Deguchi, T. 2000 *Characteristic Length of Random Knotting for Cylindrical Self-Avoiding Polygons* Phys. Lett. **A274**, 184 - 191. Cited on: 52

[135] Simon, G. W. and Leighton, R. B. 1964 Astrophys. J. **140** 1120. Cited on: 84

[136] Soteros, C. E., Sumners, D. W. and Whittington, S. G. 1992 *Entanglement complexity of graphs in* Z^3 Math. Proc. Camb. Phil. Soc. **111**, 75 - 91. Cited on: 53

[137] Stasiak, A., Katritch, V. and Kauffman, L. H. (editors) 1998 *Ideal Knots* (Vol. 19 Series on Knots and Everything, World Scientific, Singapore) Cited on: 53

[138] Stenflo, J. O. 1973 *Magnetic Field Structure of the Photospheric Network* Solar Physics **32**, 41 - 63. Cited on: 84

[139] Stenflo, J. O. 1978 *The Measurement of Solar Magnetic Fields* Rep. Prog. Phys. **41**, 865 - 907. Cited on: 84

[140] Sturrock, P. A. 1980 *Flare Models* in *Solar Flares* ed. by Sturrock, P. A. (Colorado Associated University Press, Boulder, CO, USA) 411 - 449. Cited on: 85

[141] Sumners, D. W. 1990 *Untangling DNA* Math. Intelligencer **12**, 71 - 80. Cited on:

[142] Tait, P. G. 1877 *Some Elementary Properties of Closed Plane Curves* Messenger of Mathematics, No. 69 also in *Scientific Papers* Vol. 1 (Cambridge Uni. Press, London, 1898), 270 - 272. Cited on: 12

[143] Tait, P. G. 1877 *On Knots I* Trans. Roy. Soc. Edinburgh **28**, 145 - 190. Also in *Scientific Papers* Vol. 1 (Cambridge Uni. Press, London, 1898), 273 - 317. Cited on: 13, 52

[144] Tait, P. G. 1884 *On Knots II* Trans. Roy. Soc. Edinburgh **32**, 327 - 342. Also in *Scientific Papers* Vol. 1 (Cambridge Uni. Press, London, 1898), 318 - 334. Cited on:

[145] Tait, P. G. 1885 *On Knots III* Trans. Roy. Soc. Edinburgh **32**, 493 - 506. Also in *Scientific Papers* Vol. 1 (Cambridge Uni. Press, London, 1898), 335 - 347. Cited on:

[146] Tatsuoka, K. 1987 *Geodesics in the braid group* preprint, Dept. of Mathematics, University of Texas at Austin. Cited on: 41

[147] Thistlethwaite, M. B. 1985 *Knot Tabulations and Related Topics* in *Aspects of Topology in Memory of Hugh Dowker 1912 - 1982* LMS Lecture Notes No. 93 (Cambridge University Press, Cambridge), 1- 76. Cited on: 13, 16

[148] Threlfall, W. 1949 *Knotengruppe und Homologieinvarianten* (in German), Sitz. Heidelberger Akad. Wiss., math.-nat. Kl., 355 - 370. Cited on: 15

[149] Tourlakis, G. J. 1984 *Computability* (Reston, Reston). Cited on: 28

[150] Turing, A. M. 1937 *On Computable Numbers, with an Application to the Entscheidungsproblem* Proc. London Math. Soc. Ser. 2 **42**, 230 - 265. Cited on: 29

[151] Vogel, P. 1990 *Representations of links by braids: A new algorithm* Comment. Math. Helvetici **65**, 104 - 113. Cited on: 75

[152] Wadati, M., Deguchi, T. and Akutsu, Y. 1989 Phys. Rep. **180**, 247 - 332. Cited on: 52

[153] Waldhausen, F. 1968 *On Irreducible 3-manifolds which are Sufficiently Large* Ann. Math. **87**, 56 - 88. Cited on: 14

[154] Waldhausen, F. 1968 *The Word Problem in Fundamental Groups of Sufficiently Large Irreducible 3-manifolds* Ann. Math. **88**, 272 - 280. Cited on: 14

[155] Warren, H. P. 1999 *Measuring the Physical Properties of the Solar Corona: Results from SUMER/SOHO and TRACE* Solar Physics **190**, 363 - 377. Cited on: 83

[156] Watko, J. A. and Klimchuk, J. A. 2000 *Width Variations along Coronal Loops observed by TRACE* Solar Physics **193**, 77 - 92. Cited on: 84

[157] Woltjer, L. 1958 Proc. Natl. Acad. Sci. **44**, 489 and 833. Cited on: 84

[158] Wright, A. and Berger, M. A. 1989 *The Effect of Reconnection upon the Linkage and Interior Structure of Magnetic Flux Tubes* J. Geophysical Research **94**, 1295 - 1302. Cited on: 87

[159] Xu, P. J. 1992 *The genus of closed 3-braids* J. Knot Theory Ramifications **1**, 303 - 326. Cited on: 41

[160] Yan, Y. and Sakurai, T. 2000 *New Boundary Integral Equation Representation for Finite Energy Force-Free Magnetic Fields in Open Space above the Sun* Solar Physics **195**, 299 - 319. Cited on: 84

[161] Yoder, M. A. 1995 *String Rewriting Applied to Problems in the Braid Groups* unpublished Ph.D. thesis (Uni. South Florida). Cited on: 31, 38

[162] Yamada, S. 1987 *The minimal number of Seifert circles equals the braid index of a link* Invent. Math. **89**, 347 - 356. Cited on: 75

[163] Zirker, J. B., Martin, S. F., Harvey, K. and Gaizauskas, V. 1997 *Global Magnetic Patterns of Chirality* Solar Physics **175** 27 - 44. Cited on: 84

Index

2-simplex, 14
3-ball, 67
3-manifold, 14
  classification, 13, 14, 21
3-satisfiability, 44
3-simplex, 14

Adian-Rabin theorem, **14**
Alexander polynomial, 15
Alexander's theorem, **17**, 20, 75, 77, 79, 80
algebraic algorithm, 61–63, 65, 66
algebraic link problem, *see* Markov problem
algebraic minimization, 59
algorithm
  approximation, 43
  efficient, **42**
  inefficient, **42**
  intractable, **42**
  linear, **42**
  polynomial-time, **42**
  quadratic, **42**
  random, 43
ambient isotopy, **12**
atomic theory, 11

basic polyhedron, 16
Berger's algorithm, 50, 60
Birkhoff's theorem, **28**
Birman's conjecture, **21**
braid, **54**, 67, *see* random braid
  σ_i, **17, 18**
  σ_i^{-1}, **17, 18**
  algebraic, 52, 53, **53**, 54
  Artin generators, **17**
  ascending, **18**
  band-generator presentation, 41
  cable, 45, 46
  cables, 45, **45**
  closed, 67, 75–77, 80
  closure, **16, 17**, 25, 27
  conjugate, **19**, 20, 21
  descending, **18**
  exponent sum, **18**, 21, 48
  from knot, 79
  fundamental, 19, 48
  fundamental word, **18**, 22
  geometric, 52, 54, 55
  group, **17**, 18, 53, **59**
  center, 22, 51
  center (other pres.), **19**
  center generator, **18**
  conjugacy problem, **19**
  generators, **17**
  left-cancelative, 48
  left-cancellative, 51
  other presentation, **19**
  quotients, 21
  relations, **17**
  right-cancellative, 51
  word problem, **19**
  identity, 45
  isotopy, 19
  labeling, 25–27
  length, 34, 41, 45
  minimum
    elastic, 52
    heuristic, 52
  negative, **17**
  oriented, 17
  positive, **17**, 24, 51
  prime, **19**
  reverse, **18**
  reverse operator, **18**
  split, **19**
  weft, **45**, 46
  weft braid, 45, **45**, 46, 47
  weft form, **45**, 46, 47
  wire, 46
  wires, **45**, 46
braid index, 13
bridge number, 13

Cayley diagram, **19**, 48, 49, 51
Church-Turing thesis, 28
combinatorial problem, 42
  instance, 42
  intractable, 43
  polynomially transformable, 43
complexity, 33, 54
  average-case, **43**
  class NP, **42**, 47
  class NPC, **43**
  class P, **42**
  polynomial-time, **42**
  worst-case, **42**, 43
conjugacy move, 20

conjugacy problem, 14, 19, 21, 24, 28, **28**, 30, 33–35, 38–40
  Garside algorithm, 21
continued fraction, 69
Conway's basic theorem, **69**
critical pair lemma, **29**, **36**
  cyclic, **36, 37**
crossing number, **55**, 65
  minimum, 11, 13, 65
cyclic permutable, 34, **34**
cyclic permutation, **26**, **33**, 34
cyclic word, **34**, 35

decision problem, **42**
Diamond lemma, *see* Newman's lemma
DNA, 52

elastic
  energy, 52
  forces, 52
  relaxation, 52
embedding, 11, 14
energy, **53**, 63, **63**, *see* knot energy
  functionals, 53
  minimum, 65
equilibrium, 53
equivalence move, 20
exchange move, 21

far commutation, 60
feedback edge set, 44
fluid
  knotted, 53
force
  constrained, 57, **58**, 61–63, 65
  crossing number, 61–63, 65, 66
    feature, 63
  crossing number minimizing, **57**
  curvature, 57, 58, **59**, 61–63, 65, 66
  elastic, 57
  repulsive, **57**, 62
fundamental group, 14, 15

Garside
  exponent, **19**
  normal form, 50, 51
  remainder, **19**
group, 28
  center, 24
  combinatorial theory, 28
  conjugacy class, 30, 39
  conjugate, 24, **33**, 34, 38
  equivalence class, 30
  free, **33**, 41
  free products, 41
  HNN-extensions, 41
  infinite cyclic, 26
  isomorphism, 14, 15
  presentation, 28
grouping by swapping, 44, **44**
Grzegorczyk arithmetical hierarchy, *see* Kleene arithmetical hierarchy

Haken, 13
Hamiltonian circuit, **78**
Hemion's algorithm, *see* 3-manifold classification
heuristic algorithm, 60
Higman Embedding Theorem, **28**
homeomorphism, 14
Homfly polynomial, 13
Hopf link, **12**, 16, 25

ideal knot, *see* knot
invariant, 13, *see* energy
  bounded, **53**
  complete, **13**, 14, 21
  component number, 13
  Gauss linking integral, 53
  helicity integral, 53
  incomplete, **13**
  isotopy, **52**
  minimum, 13
  minimum crossing number, 53
  polynomial, 13
  polynomials, 53
invariants
  minimum crossing number, 55
inversion, 44, **44**, 45–47
inversions, 44
isomorphism, 14
isomorphism problem, 28, **28**

Jones polynomial, 13, 16

Kleene arithmetical hierarchy, 28
knot, **11**, **14**, 80, 81, *see* random knot
  2-bridge, 69
  amphicheiral, 21
  classification, 11–13, 15, 21
  complement, 13, 14, **14**, 25, 27
  component, 15, 25, 73, 74
  composite, **13**, 21
  diagram, 13
  energy, 53
  fundamental group, 25, 27
  group, **15**
    Wirtinger presentation, **15**
    Wirtinger representation, 25, 27
  ideal, **53**, 65
  invertible, 21
  isotopy, 15
  isotopy problem, 13
  longitude, 15, **15**, 25
  matrix form, **70**
  meridian, 15, **15**, 25

meridian-longitude system, **15**, 25
trivial, **26**
nugatory crossing, **20**
oriented, 72
peripheral group system, 25
prime, 11, **13**, 21
factorization, 21
split, 21
sum, **13**, **73**
table, 11, 12, 16, 52
type, 52
knot notation, *see* knotation
knotation, **16**, 69, 71
Conway, 16, 67, 72, 80, 81
Dowker & Thistlethwaite, 16
Knuth-Bendix completion, **29**, 30, 38

L-move, 20
Lagrangian, 56, 59, 65
lexicographic order, 19, 39
link, *see* knot
linking number, **14**

magnetic field, 52
accretion disk, 52
continuous, 53
coronal loop, 53
lines, 65
reconnection, 53
star, 52
Markov, 20
Markov equivalence, 20, **20**, 21, 25
Markov move, 20, **20**, 21
Markov problem, 20, **20**, 21, 24
Markov's theorem, 20, **20**, 21, 80
Michelson and Morley, 11
minimal equivalent braid, **45**
minimization problem, **41**
minimum
global, 53, 63
local, 53, 63, 65
minimum crossing number, *see* crossing number, *see* invariant
minimum string number, 21
minimum word problem, 51
monoid, 28, 38

Newman's lemma, **29**, 36, **36**
non-minimal braids, **44**, 47
NP-complete, 28, 41, 60, 78
NP-completeness, 42, 43, **43**, 45, 47
reduction, 43
restriction, 44, **44**

peripheral group system, 14, **15**
permutation, 44–47
identity, 44
plait, 67, 76
closed, 76
from knot, 78
polyhedron, **69**, 71, 76, 79
axis, 77, 79
basic, **69**, 80
matrix, **70**, 72, 80
edge, **70**
vertex, **70**
region, 73, 74
universal, **70**
polymer, 52
polynomials, 52
program, **42**

random braid, 52–54, 61, 65
random knot, 52
recursive set, **28**
recursively enumerable set, **28**
reduction ratio, 60–62
Reidemeister, 13
moves, 12, 13, 17, 20
relaxation, 55, 57

satisfiability, 43, 44
search problem, **45**
shortlex ordering, 39
simulation
efficacy, 61
efficiency, 62
skein relation, 13
solar flare, 53
solar physics, 53, 65
sorting does not minimally partition, 44, **44**, 47
stabilization, *see* Markov move
statistical mechanics
exactly solvable model, 52
surgery, **12**

Tait, 11, 52
tangle, 16, 67, **67**
classification, 69, **69**
elementary, 68, **68**, 69, 81
equality, **67**
fractional, 68, **68**, 69, **69**, 70
integral, 68, **68**, 69, 80, 81
irrational, 68, **69**
rational, 68, **68**, 69, 70
sum, **68**
tangles, 69
elementary, 72
term rewriting systems, *see* trs
tetrahedron, 14
topological complexity
measure of, 52
topology
change, 57

torus knot, 15
trefoil, 71, **71, 72**
triangulation, 14
trs, 27, **27**, 28, 35
  c-obstruction, **35**, 36, 38
  complete, **28**, 29, 31, 38
  confluence, **28**, 29, 35, 36, 40
    undecidability, 29
  constants, **27**
  critical pair, 29, **29**, 32, 36, 37, 40
  cyclic completeness, **35**, 38
  cyclic confluence, **35**, 36, 40
  cyclic critical pair, **36**, 37, 40
  cyclic local confluence, **35**, 36–38
  cyclic overlap, **36**, 39
    critical, **36**
    non-critical, **36**
  cyclic termination, 35, **35**, 36, 38, 40
  cyclic word, 35, 37, 39
  equivalence, 32
  final form, 38
  joinability, **27**
  length metric, **35**
  length reducing, **35**
  local confluence, **28**, 32
  normal form, 28
  overlap, **29**, 32, 36
    critical, **29**
    non-critical, **29**
  redex, **28**, 36, 37
  reduct, **28**, 32, 37
  reduction order, 29, **29**, 30, 35
    basis, **29**
    closure, **29**
    compatibility, **29**
  rewrite chain, **28**
  rule, **27**
  substitution, **27**
  term, **27**
  termination, **28**, 29, 32, 35, 36, 38
    undecidability, 29
  total order, 35
  weight metric, **35**
  weight reducing, **35**
  word, **27**
Turing Halting Problem, 29, 42
Turing machine, 28, 29, 42
  deterministic, **42**
  non-deterministic, **42**

unknot, **12**, 13, **77**
  recognition, 13
unknotting number, 16

vertex cover, 44
vortex lines, 52

Waldhausen's theorem, **14**
Whitehead link, **12**
word problem, 14, 19, 21, 24, 28, **28**, 29–31, 33, 38, 39, 54

Yoder's theorem, 38, **38**