NANYANG TECHNOLOGICAL UNIVERSITY

On the Genus Expansion of the Colored HOMFLY Polynomial

by

Vo Thanh Huan

A thesis submitted as part of the honors requirements

in the

Division of Mathematical Sciences

School of Physical and Mathematical Sciences

November 2009

NANYANG TECHNOLOGICAL UNIVERSITY

Abstract

Division of Mathematical Sciences
School of Physical and Mathematical Sciences

by Vo Thanh Huan

This thesis aims to give a brief overview about the recent developments in the theory of quantum link invariants, in particular the colored HOMFLY polynomial. A method to calculate the colored HOMFLY polynomial via the cabling-projection rule was discovered recently, which allows implementation on a computer. It turns out that the irreducible characters of the Hecke algebra play a crucial role in the computation of the colored HOMFLY polynomial. A computer program is written to output the colored HOMFLY polynomial of oriented links, which helps verify certain new cases of the Labastida-Mariño-Ooguri-Vafa (LMOV) conjecture.

Acknowledgements

I would like to thank my supervisor, Asst Prof Andrew James Kricker, for his patient guidance and willingness to answer my questions regarding the subject. His knowledge has benefited me greatly throughout the project. I would also like to thank all the staff in the Mathematics Department of SPMS for their help during my undergraduate education. Finally, I thank my parents for their endless support.

Contents

List of Figures

Chapter 1

Introduction

The mathematical theory of knots has experienced important progress in recent years. By a *knot* (*link*), we mean a smooth embedding of one (several) copies of the circle S^1 into the three-dimensional space $\mathbb{R}^3$. Intuitively, a knot is just a knotted loop of string in space that does not intersect itself anywhere, except that we think of the string as having no thickness, its cross-section being a single point. The string is thought of as if it were made of easily deformable rubber. Thus we can twist, bend or pull the string to get different pictures of the same knot (of course the string is not allowed to be cut or pass through itself). We call such a deformation of a knot an *ambient isotopy* in mathematical terms. We will consider two knots to be the same (isotopic) if they are related by an ambient isotopy of the three-dimensional space.

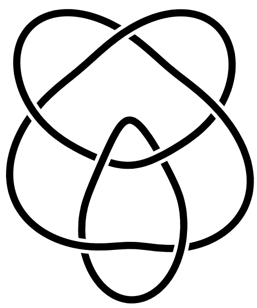


FIGURE 1.1: A two-component link.

By projecting a knot in the two-dimensional plane, we obtain a *projection* of the knot. The *unknot* (or *trivial knot*) is the knot which has a projection which resembles the circle. One of the main problems in knot theory is to decide whether two different projections represent the same knot, or a simpler version whether a projection represents the unknot. Interestingly enough, this problem was motivated mainly by chemistry, dated back in the nineteenth century. In the 1880s, the English physicist William Thomson (a.k.a., Lord Kelvin) hypothesized that atoms were merely knotted vortices in the ether and that different knots would then correspond to different elements. This led the Scottish physicist Peter Guthrie Tait to the quest of listing all distinct knots. Unfortunately, Kelvin was wrong. A more accurate model for the atomic structure appeared at the end of the nineteenth century and chemists lost interest in knots for the next 100 years. But in the meantime, mathematicians had become intrigued with knots and so a century of work on the mathematical theory of knots followed.

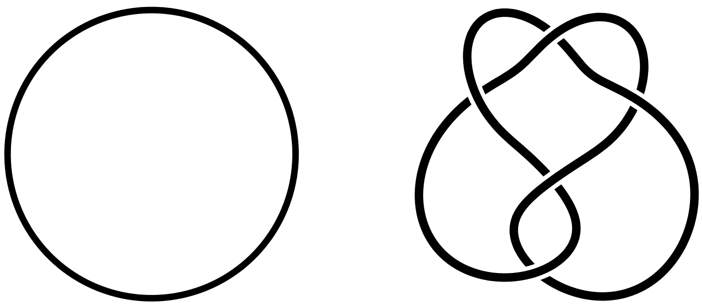


FIGURE 1.2: Are these two projections of the same knot?

To show that two projections represent the same knot, we just need to present a smooth deformation which transforms one projection into the other. However, the task of showing two projections represent two distinct knots is not so simple. Just because we played with one projection for a long time and could not transform it to the other does not mean that these two are different. (In fact there are instances of knots thought to be distinct for decades were discovered to be the same, see Perko pair [16].) In 1926, the German mathematician Kurt Reidemeister proved that two projections represent the same knot if and only if they are related by a finite sequence of *Reidemeister moves* (Figures 1.3, 1.4 and 1.5) and planar isotopy (deforming the plane on which the knot is drawn).

A knot (link) invariant is some quantity associated to a projection which is unchanged under the three Reidemeister moves. Thus it is a well defined function on the isotopy classes of knots. In other words, if two projections have different knot invariants, then they represent different knots (the converse is not true

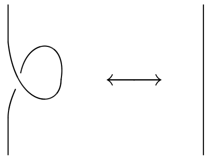


FIGURE 1.3: The first Reidemeister move.

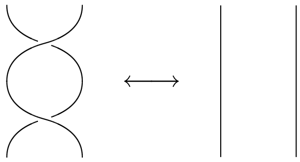


FIGURE 1.4: The second Reidemeister move.

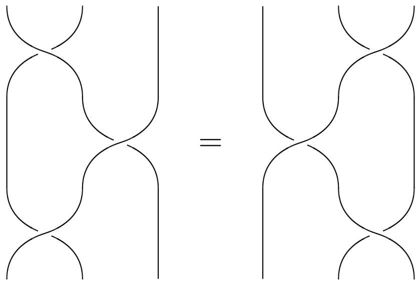


FIGURE 1.5: The third Reidemeister move.

however) and so we have a tool to tell knots apart. Many link invariants were discovered from different points of view. Each one has their strength and weaknesses in distinguishing knots. In this thesis, we will focus on one particular type of link invariants: the polynomial link invariants.

The first use of polynomial link invariants was due to J.Alexander in about 1928 with his Alexander polynomial [1] (actually, it is a Laurent polynomial, so we allow negative powers of the variable). Mathematicians utilized the Alexander polynomial to distinguish knots and links for the next 50 years, until the discovery of the Jones polynomial. In 1984, the New Zealand mathematician Vaughan Jones discovered a new polynomial for knots and links, which came out of work he was doing on operator algebras. The Jones polynomial is a powerful knot invariant. Indeed, the question of whether there is any non-trivial knot with the same Jones polynomial as that of the unknot is still open (the answer is yes for the case of links). Jones's discovery generated immense excitement among mathematicians. Four months after the announcement of the Jones polynomial, the HOMFLY polynomial was announced, which was a generalized version of the former. (The name HOMFLY came from the first letters of the names of the discoverers.) It didn't take mathematicians a long time to realize that these polynomials were all special cases of a much bigger picture. A general procedure was discovered that could produce a whole family of similar polynomials, using some mathematical "machine" called a quantum group. We call a polynomial link invariant obtained in this manner a quantum link invariant. In particular, the quantum link invariant obtained from the quantum group $U_q(sl_N)$ is called the *colored HOMFLY polynomial*, which is the main focus of this thesis.

In general, it is quite difficult to compute the colored HOMFLY polynomial directly from definition. Nevertheless, there is a more practical method to calculate the colored HOMFLY polynomial via the cabling-projection rule [9]. The formula obtained involved irreducible characters of the Hecke algebra. Using the combinatorial algorithm described in [12], we were able to write a computer program to calculate the irreducible characters of the Hecke algebra and subsequently the colored HOMFLY polynomial of oriented links. This allows us to test certain cases of the Lasbatida-Mariño-Ooguri-Vafa (LMOV) conjecture. This conjecture arises from a conjectured relationship between the $1/N$ expansion of Chern-Simons thoery and the Gromov-Witten invariants of certain non-compact Calabi-Yau threefolds.

The thesis is organized as follows. In chapter 2, we review some basic facts about representation theory of groups and algebras. We also introduce the notion of Young tableaux and Schur polynomials. In chapter 3, we present a detailed algorithm to compute the irreducible characters of the Hecke algebra, with the Mathematica codes included. In chapter 4, we define the colored HOMFLY polynomial and describe a way to compute them algorithmically. Finally, in chapter 5 we introduce briefly the LMOV conjecture and test it against certain cases of the figure 8 knot.

Chapter 2

Background on Representation Theory

Throughout the thesis, we will extensively use facts concerning the representation theory of algebras. In this section, we review some basic terminologies in representation theory. We also introduce the notions of Young diagrams and Schur polynomials.

2.1 Algebras

Let k be an algebraically closed field, i.e. any nonconstant polynomial with coefficients in k has a root in k. In this thesis, we will mainly deal with the field of complex numbers $\mathbb{C}$.

Definition 2.1. An *algebra* over k is a vector space A over k together with a bilinear map $A \times A \to A$, $(a, b) \mapsto ab$, which satisfies

(i) $(\mu a + b)c = \mu ac + bc$, (linearity in the first position),

(ii) $a(\mu b + c) = \mu ab + ac$, (linearity in the second position),

(iii) $(ab)c = a(bc)$, (associativity),

for all $a, b, c \in A$ and $\mu \in k$.

Definition 2.2. A *unit* in an algebra A is an element $1 \in A$ such that $1a = a1 = a$ for all $a \in A$.

Remark 2.3. If a unit exists, then it is unique. Thus we can speak of the unit in an algebra.

From now on, by an algebra A we will mean a finite-dimensional algebra over k with the unit.

Example 2.1. Here are some examples of algebras over k

1. $A = k$.

2. $A = \mathrm{End}(V)$, the algebra of all endomorphisms of a vector space V over k (linear maps from V to itself). The multiplication is given by composition of operators.

3. The *free algebra* $A = k\langle x_1, ..., x_n\rangle$ has as its basis finite words in letters $x_1, ..., x_n$. Multiplication of words is given by concatenation.

4. The *group algebra* $A = k[G]$ of a finite group G. Its basis is $\{a_g, g \in G\}$ with multiplication law $a_g a_h = a_{gh}$.

Definition 2.4. A *homomorphism* of algebras $f: A \to B$ is a linear map such that $f(xy) = f(x)f(y)$ for all $x, y \in A$ and $f(1) = 1$.

Definition 2.5. A *left ideal* of an algebra A is a subspace I of A such that $ax \in A$ for all $a \in A$ and $x \in I$. Similarly, a *right ideal* of an algebra A is a subspace of I such that $xa \in A$ for all $a \in A$ and $x \in I$. A *two-sided ideal* is a subspace which is both a left and a right ideal.

Example 2.2. Here are some examples of ideals.

1. If A is any algebra, then 0 and A are two sided-ideals.

2. If $\phi: A \to B$ is an algebra homomorphism, then $\ker\phi$ is a two-sided ideal of A.

3. If S is any subset of an algebra A, then the two-sided ideal *generated* by S, denoted $\langle S\rangle$, is the smallest two-sided ideal which contains S. It is the span of elements of the form asb, where $a, b \in A$ and $s \in S$.

Definition 2.6. Let A be an algebra and I a two-sided ideal of A. We can make the quotient vector space A/I into an algebra by defining multiplication on the cosets of I as follows

$$(a + I)(b + I) = ab + I, \quad a, b \in A.$$

It can be verified that this multiplication is well-defined. The resulting algebra is called the *quotient algebra* A/I.

Remark 2.7. If $f_1, ..., f_m$ are elements of the free algebra $k\langle x_1, ..., x_n\rangle$, we say that the algebra $A = k\langle x_1, ..x_n\rangle / \langle\{f_1, ..., f_m\}\rangle$ is *generated by* $x_1, ..., x_n$ *with defining relations* $f_1 = 0, ..., f_m = 0$.

2.2 Representations of Algebras

Definition 2.8. A *representation* of an algebra A is a vector space V over k together with a homomorphism of algebras $\rho\colon A \to \mathrm{End}(V)$ (i.e. ρ is a linear transformation which satisfies $\rho(ab) = \rho(a)\rho(b)$ for all $a, b \in A$). The vector space V is also called a *left* A*-module*.

Remark 2.9. If (V, ρ) is a representation of A, we often write $\rho(a)(v)$ as $a \cdot v$ for $a \in A$ and $v \in V$ and speak of the *action of* a *on* V. In this notation, we have

$$(ab) \cdot v = a \cdot (b \cdot v), \quad a, b \in A,\ v \in V.$$

Example 2.3. We give some representations of an algebra A.

1. $V = 0$.

2. $V = A$, and $\rho\colon A \to \mathrm{End}A$ is defined as follows

   $$\rho(a)(b) = ab, \quad a, b \in A.$$

   Thus the action of $a \in A$ is the left multiplication in the algebra. The linearity of ρ follows from the linearity of the multiplication. Furthermore,

   $$\rho(ab)(c) = (ab)c = a(bc) = \rho(a)\rho(b)(c), \quad c \in A,$$

   by associativity. This representation is called the (left) *regular* representation of A.

Definition 2.10. A *subrepresentation* of a representation V of an algebra A is a subspace W of V which is closed under the action of the algebra, i.e. $a \cdot w \in W$ for all $a \in A$ and $w \in W$.

Example 2.4. Recall that A can be considered as a representation of itself under the regular representation. In this case, all left ideals of A are subrepresentations of the regular representation of A.

Definition 2.11. A representation $V \neq 0$ of A is called *irreducible* (or *simple*) if the only subrepresentations of V are 0 and V.

Definition 2.12. Let V_1, V_2 be two representations of an algebra A. A *homomorphism of representations* (or *intertwiner*) $\phi\colon V_1 \to V_2$ is a linear transformation which commutes with the action of A, i.e. $\phi(a \cdot v) = a \cdot \phi(v)$ for any $v \in V_1$. A homomorphism ϕ is said to be an isomorphism of representations if it is an isomorphism of vector spaces. The set of all homomorphisms of representations $V_1 \to V_2$ is denoted by $\mathrm{Hom}_A(V_1, V_2)$.

Remark 2.13. Let V_1, V_2 be representations of an algebra A. The space $V = V_1 \oplus V_2$ can be made into a representation of A by

$$a \cdot (v_1 \oplus v_2) = a \cdot v_1 \oplus a \cdot v_2, \quad a \in A,\ v_1 \in V_1,\ v_2 \in V_2.$$

The representations V_1 and V_2 can be identified with subrepresentations of V in a natural way.

Two representations between which there exists an isomorphism are said to be isomorphic. For practical purposes, two isomorphic representations may be regarded as "the same", although there could be subtleties related to the fact that an isomorphism between two isomorphic representations may not be unique.

Definition 2.14. A representation $V \neq 0$ of an algebra A is said to be *indecomposable* if it is not a direct sum of two nonzero representations.

Remark 2.15. It's clear that an irreducible representation is indecomposable. However, the converse is not true in general.

The following theorem is fundamental in the whole subject of representation theory.

Theorem 2.16 (Schur's lemma)**.** *Let V, W be finite-dimensional irreducible representations of an algebra A over any field F (which needs not be algebraically closed), and $\phi\colon V \to W$ is an intertwiner. Then $\phi = 0$ or ϕ is an isomorphism.*

Remark 2.17. If $V = W$ and $F = \mathbb{C}$ and $\phi \neq 0$, then $\phi = \lambda \mathrm{id}$ for some $\lambda \in k$ (a scalar multiplication of the identity map).

2.3 Representations of Groups

All vector spaces considered in this section are finite-dimensional over $\mathbb{C}$ and all groups are finite.

Definition 2.18. Let V be an n dimensional complex vector space and G a finite group. Let $\mathrm{GL}(V)$ be the group consisting of all invertible linear transformations of

V under function composition. A representation of G is a vector space V together with a group homomorphism $\rho\colon G \to \mathrm{GL}(V)$. If we choose a basis for V, the group $\mathrm{GL}(V)$ can be identified with the group of $n \times n$ invertible matrices over $\mathbb{C}$, $M_n(\mathbb{C})$. Thus we can think of representation as a way of assigning invertible matrices to elements of the group.

Remark 2.19. Recall that the group algebra $\mathbb{C}[G]$ of a finite group G consists of all formal sums of the form

$$\sum_{g\in G} \alpha_g g, \quad \alpha_g \in \mathbb{C}.$$

Given a representation (V, ρ) of G, we can make V into a (left) $\mathbb{C}[G]$-module as follows

$$\left(\sum_{g\in G} \alpha_g g\right) \cdot v = \sum_{g\in G} \alpha_g \rho(g)(v), \quad v \in V.$$

Conversely, given a $\mathbb{C}[G]$-module V, we can get a representation V of G by

$$\rho(g)(v) = g \cdot v, \quad g \in G,\ v \in V.$$

Thus giving a representation V of G is equivalent to giving a $\mathbb{C}[G]$-module V and so all the definitions for representations of algebras apply to representations of groups. More details can be found in [3].

Example 2.5. We give some examples of representations of a group G

1. Let $V = \mathbb{C}[G]$. Recall that V can be thought of as a $\mathbb{C}[G]$-module under left multiplication and thus is a representation of G. We call V the *regular representation* of G.

2. Suppose G acts on a finite set X, i.e. for each $g \in G$, there is given a permutation $x \mapsto g \cdot x$ of X satisfying the identity

$$1 \cdot x = x, \quad g \cdot (h \cdot x) = (gh) \cdot x, \text{ for } x \in X,\ g, h \in G.$$

Let V be a vector space having a basis $(e_x)_{x\in X}$ indexed by the elements of X. We can make V into a representation of G by

$$g \cdot e_x = e_{g\cdot x}, \quad g \in G,\ x \in X.$$

This representation is called the *permutation representation* associated with X.

Theorem 2.20. *Let V be a representation of G and W is a subrepresentation of V. Then there exists a subrepresentation Z of V such that $V = W \oplus Z$.*

Corollary 2.21. *Every representation of G is a direct sum of irreducible representations.*

Definition 2.22. Let $\rho\colon G \to \mathrm{GL}(V)$ be a representation of G. The *character* of ρ is a function $\chi_\rho\colon G \to \mathbb{C}$ given by

$$\chi_\rho(g) = \mathrm{tr}(\rho(g)),$$

where $\mathrm{tr}(f)$ denotes the usual trace of a linear transformation f (sum of the diagonal entries).

Remark 2.23. By basic linear algebra, we have

$$\chi_\rho(ghg^{-1}) = \chi_\rho(h), \text{ for all } g, h \in G.$$

Thus χ_ρ can be considered as a function on conjugacy classes of G. For this reason, it is called the *class function*.

Theorem 2.24. *The irreducible representations of G (up to isomorphism) are indexed by conjugacy classes of G.*

2.4 Young Tableaux and Schur Polynomials

The notion of Young tableaux plays a crucial role in the representation theory of the symmetric group. We know that irreducible representations of a finite group G are indexed by conjugacy classes of G. For the case of the symmetric group S_n, the conjugacy classes are naturally indexed by *partitions* of n, i.e., a sequence of non-increasing positive integers whose sum is n. A partition of n can be presented in a form of a *Young diagram*, as in Figure 2.1.

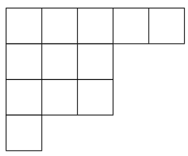


FIGURE 2.1: The partition $(5, 3, 3, 1)$ of 12.

We usually denote a Young diagram by a lowercase Greek letter, such as λ. The notation $\lambda \vdash n$ is used to say that λ is a partition of n, and $|\lambda|$ is used for the number partitioned by λ. We also use $l(\lambda)$ to denote the number of positive integers in λ. Any way of putting a positive integer in each box of a Young diagram will be called a *filling* of the diagram. A *Young tableau* (or *tableau* for short) (Figure 2.2) is a filling that is

1. weakly increasing across each row,

2. strictly increasing down each column.

A *standard tableau* (Figure 2.2) is a tableau in which the entries are the numbers from 1 to n, each occurring once.

$$\begin{array}{|c|c|c|c|}\hline 1 & 1 & 2 & 2 \\ \hline 3 & 4 & 4 \\ \cline{1-3} 4 & 5 \\ \cline{1-2}\end{array} \qquad \begin{array}{|c|c|c|c|}\hline 1 & 3 & 4 & 8 \\ \hline 2 & 5 & 6 \\ \cline{1-3} 7 & 9 \\ \cline{1-2}\end{array}$$

FIGURE 2.2: A tableau and a standard tableau on $(4, 3, 2)$.

A *skew digram* (Figure 2.3) is a diagram obtained by removing a smaller Young diagram from a larger one that contains it. If two diagrams correspond to partitions $\lambda = (\lambda_1, \lambda_2, ...)$ and $\mu = (\mu_1, \mu_2, ...)$, we write $\mu \subset \lambda$ if the Young diagram of μ is contained in that of λ; equivalently, $\mu_i \leq \lambda_i$ for all i. The resulting skew diagram is denoted λ/μ.

$$\begin{array}{|c|c|c|c|}\hline & & \bullet & \bullet \\ \hline & \bullet & \bullet \\ \cline{1-3} \bullet & \bullet \\ \cline{1-2}\end{array}$$

FIGURE 2.3: The skew diagram $(4, 3, 2)/(2, 1)$.

Let m be a positive integer and λ be a Young diagram with at most m rows. Let $x_1, ..., x_m$ be m commuting variables. For each Young tableau T on λ, we associate a monomial x^T, which is a product of the variables x_i corresponding to the i's that occur in T. Formally,

$$x^T = \prod_{i=1}^{m} (x_i)^{\text{number of times } i \text{ occurs in } T}.$$

The *Schur polynomial* in m variables $x_1, ..., x_m$ corresponding to the Young diagram λ, $s_\lambda(x_1, ..., x_m)$ is defined as

$$s_\lambda(x_1, ..., x_m) = \sum x^T,$$

where the sum is over all tableaux of shape λ using the numbers from 1 to m. For instance, if $m = 3$ and $\lambda = (2, 1)$, then the tableaux are

$$\begin{array}{|c|c|}\hline 1 & 1 \\ \hline 2 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array} \quad \begin{array}{|c|c|}\hline 1 & 2 \\ \hline 2 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array} \quad \begin{array}{|c|c|}\hline 1 & 3 \\ \hline 2 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array} \quad \begin{array}{|c|c|}\hline 1 & 1 \\ \hline 3 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array}$$

$$\begin{array}{|c|c|}\hline 1 & 2 \\ \hline 3 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array} \quad \begin{array}{|c|c|}\hline 1 & 3 \\ \hline 3 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array} \quad \begin{array}{|c|c|}\hline 2 & 2 \\ \hline 3 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array} \quad \begin{array}{|c|c|}\hline 2 & 3 \\ \hline 3 & \multicolumn{1}{c}{} \\ \cline{1-1}\end{array}$$

Thus the Schur polynomial is given by

$$s_\lambda(x_1, x_2, x_3) = x_1^2 x_2 + x_1 x_2^2 + x_2^2 x_3 + x_2 x_3^2 + x_1^2 x_3 + x_1 x_3^2 + 2x_1 x_2 x_3.$$

It is an interesting fact that the Schur polynomials are symmetric [6].

Chapter 3

Computing the Characters of the Hecke Algebra

It turns out that the colored HOMFLY polynomial can be expressed in terms of the irreducible characters of the Hecke algebra and Schur polynomials. In this section, we present a detailed combinatorial algorithm to calculate the characters of the Hecke algebra, with the Mathematica codes included.

3.1 The Symmetric Groups

The symmetric group S_n can be defined as the group generated by generators $s_1, s_2, \ldots, s_{n-1}$ and relations

$$\begin{cases} s_i s_j = s_j s_i, & |i-j| > 1, \\ s_i s_j s_i = s_j s_i s_j, & |i-j| = 1, \\ s_i^2 = 1, & i = 1, 2, ..., n-1. \end{cases}$$

The elements s_i are called simple transpositions. The *length*, $l(\sigma)$, of an element σ in S_n is the minimum number of simple transpositions necessary to express σ. Any product of $l(\sigma)$ transpositions equal to σ is called a *reduced decomposition* of σ. By viewing each generator s_i as the operation that switches i and $i+1$ each element of S_n can be viewed as a permutation of $1, 2, ..., n$. We write $\sigma(i)$ for the image of i under σ. We can express σ in S_n in the following permutation notation

$$\sigma = \begin{pmatrix} 1 & 2 & \cdots & n \\ \sigma(1) & \sigma(2) & \cdots & \sigma(n) \end{pmatrix} = (\sigma(1), \sigma(2), \ldots, \sigma(n)).$$

Multiplication of permutations is performed from left to right. Thus if $\mu = \sigma_1\sigma_2$ is a product of two permutations in S_n, then we have $\mu(i) = \sigma_2(\sigma_1(i))$ for all $i = 1, 2, ..., n$. For example,

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}.$$

There is an embedding of $S_m \times S_p$ into S_{m+p}, $(\sigma, \pi) \longmapsto \sigma \times \pi$, given by making S_m act on $1, 2, ..., m$ and S_p act on $m+1, m+2, ..., m+p$. For instance,

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 1 & 5 & 6 & 4 \end{pmatrix}.$$

The r-*cycle* is the element

$$\gamma_r = s_{r-1}s_{r-2} \cdots s_2 s_1$$

of S_r. The *1-cycle* is the identity $1 \in S_1$. A *composition* c of n, denoted by $c \models n$, is a sequence of positive integers whose sum equals n. For each $c = (c_1, c_2, ...)$, composition of n, we define $\gamma_c \in S_n$ by

$$\gamma_c = \gamma_{c_1} \times \gamma_{c_2} \times \cdots .$$

Given any permutation $\sigma \in S_n$, there exists some permutation π such that $\pi\sigma\pi^{-1} = \gamma_\lambda$ for some partition λ of n. The partition λ is the *cycle type* of the permutation σ. Any two permutations with the same cycle type are said to be in the same conjugacy class.

It is a standard result that the irreducible representations of S_n are indexed conjugacy classes of S_n and thus by partitions of n. The dimension of the irreducible representation indexed by the partition λ will be denoted d_λ, and the irreducible character determined by this representation by χ^λ.

3.2 The Hecke Algebras

Let $\mathbb{C}(q)$ be the field of rational functions in the variable q. The Hecke algebra $H_n(q)$ of type A_{n-1} is the $\mathbb{C}(q)$ algebra generated by $g_1, g_2, ..., g_{n-1}$ and relations

$$\begin{cases} g_i g_j = g_j g_i & |i-j| \geq 2, \\ g_i g_j g_i = g_j g_i g_j, & |i-j| = 1, \\ g_i^2 + (q^{-1} - q) g_i - 1 = 0, & i = 1, ..., n-1. \end{cases}$$

For each $\sigma \in S_n$, let $T_\sigma = g_{i_1} g_{i_2} \cdots g_{i_k}$, where $\sigma = s_{i_1} s_{i_2} \cdots s_{i_k}$ is a reduced decomposition of σ. It is well known that each element T_σ is independent of the reduced decomposition of σ and that the set of elements T_σ, $\sigma \in S_n$ form a basis of $H_n(q)$. Thus $H_n(q)$ has dimension $n!$. Note that if we specialize $q = 1$, the third relation becomes $g_i^2 = 1$, which is the same as that of the symmetric group.

From another point of view, the Hecke algebra $H_n(q)$ is just the group algebra of the braid group B_n with one additional relation, by viewing each generator g_i as the positive crossing σ_i. The element T_σ for $\sigma \in S_n$ is also called a *positive permutation braid* (Figure 3.1), which is defined as a braid where every two strands cross at most once positively.

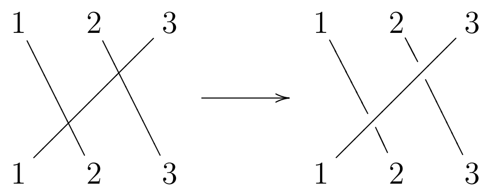


FIGURE 3.1: The elements γ_3 and T_{γ_3}.

The irreducible reprerentations of $H_n(q)$ are also indexed by partitions of n [11]. For each partition λ of n, the dimension of the irreducible representation indexed by λ is d_λ as in the case of S_n, and determines an irreducible character ζ^λ of $H_n(q)$.

Our goal is to compute the irreducible characters ζ^λ of the the Hecke algebra. For this purpose, we first describe a combinatorial algorithm, which is the q-extension of the Murnaghan-Nakayama rule, to compute the characters on the special elements T_{γ_μ}, where μ is a partition of n. For simplicity of notations, we write

$$\zeta^\lambda(T_{\gamma_\mu}) = \zeta^\lambda(\mu).$$

Given a partition $\mu = (\mu_1, \mu_2, ..., \mu_f)$ of n, recall that T_{γ_μ} (Figure 3.2) is obtained from $\gamma_\mu = \gamma_{\mu_1} \times \gamma_{\mu_2} \times \cdots \times \gamma_{\mu_f}$ by replacing simple transpositions in the reduced decomposition of γ_μ by positive crossings.

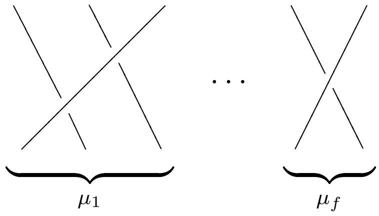


FIGURE 3.2: The elements T_{γ_μ}.

3.3 The Quantized Murnaghan-Nakayama Rule

The Murnaghan-Nakayama rule is a combinatorial algorithm to compute the characters of the symmetric group. In this section, we will present a quantized version of the Murnaghan-Nakayama rule [12] to compute the characters of the Hecke algebra. A skew diagram λ/μ is called a *broken rim hook* if it does not contain any 2×2 block of boxes. A broken rim hook is called a *rim hook* if it is connected, i.e., any two consecutive boxes share a common edge. Note that any broken rim hook is a subset of the *maximal rim hook* (in Figure 3.3, we use the "•" to denote the skew diagram), which starts at the first box of the last row of the diagram, traverses along the south-east border up to the last box in the first row. Thus a

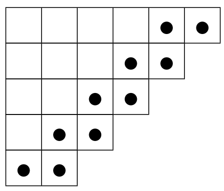


FIGURE 3.3: The maximal rim hook of $(6, 5, 4, 3, 2)$.

broken rim hook is a collection of rim hooks. For a broken rim hook b, we let $n(b)$ (see Figure 3.4) denote the number of rim hooks it contains. For a rim hook h, we define its weight $wt_q(h)$ to be

$$wt_q(h) = (-1)^{r(h)-1} q^{2(c(h)-1)},$$

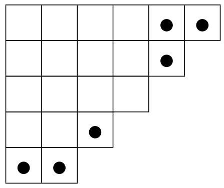


FIGURE 3.4: The broken rim hook $(6,5,4,3,2)/(4,4,4,2)$ which contains 3 rim hooks.

where $r(h)$ is the number of rows and $c(h)$ is the number of columns of h, respectively. The weight of a broken rim hook b is

$$wt_q(b) = (q^2-1)^{n(b)-1} \prod_{\text{rim hooks } h \in b} wt_q(h).$$

Given a partition $\mu = (\mu_1, \mu_2, ...\mu_f)$ of n, a *μ-broken rim hook decomposition* T of a partition λ of n is a sequence of partitions

$$\emptyset = \lambda^{(0)} \subset \lambda^{(1)} \subset \cdots \subset \lambda^{(f)} = \lambda,$$

such that for each $1 \leq i \leq f$, $\lambda^{(i)}/\lambda^{(i-1)}$ is a broken rim hook of $\lambda^{(i)}$ with μ_i boxes. The weight of each μ-broken rim hook decomposition T of λ is defined to be

$$wt_q(T) = \prod_{i=1}^{f} wt_q(\lambda^{(i)}/\lambda^{(i-1)}).$$

Finally, the quantized Murnaghan-Nakayama rule says that the character ζ^λ of the Hecke algebra $H_n(q)$ is given by

$$\zeta^\lambda(\mu) = q^{f-n} \sum_T wt_q(T),$$

where the sum is over all μ-broken rim hook decompositions of λ.

3.4 Decomposition of Young Diagrams

We shall implement the above algorithm in Mathematica. The goal is to find all μ-broken rim hook decompositions of a partition λ. To achieve this, we need several subroutines, which will be described below.

The first subroutine is called *rim* (Figure 3.5), which takes a partition τ and a box in the maximal rim hook and outputs the portion of the maximal rim hook (sequence of boxes) starting from that box. We input a partition $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k)$

as a list $\{\lambda_1, \lambda_2, ..., \lambda_k\}$ in Mathematica and $\{i, j\}$ is the position of a box, where we label the boxes from left to right, top to bottom, as in the matrix fashion. The idea is pretty straight forward: if a box is the last box of the first row, then we just output that box, otherwise if a box is not the last box of its row, we append that box to the portion of the maximal rim hook starting from the box right next to it in the same row, else we append that box to the portion of the maximal rim hook starting from the box right above it in the same column. So for example if

```
In[66]:= rim[tau_, box_] :=
       Module[{r}, If[box[[1]] == 1 && box[[2]] == tau[[1]], r = {{1, tau[[1]]}},
         If[box[[2]] < tau[[box[[1]]]],
          Join[{{box[[1]], box[[2]]}}, rim[tau, {box[[1]], box[[2]] + 1}]],
          Join[{{box[[1]], box[[2]]}}, rim[tau, {box[[1]] - 1, box[[2]]}]]]]]
```

FIGURE 3.5: The Mathematica code for rim.

$\tau = \{6, 5, 4, 3, 2\}$ and $\{3, 3\}$ is the position of the box (second row, second column), then the output will be $\{\{3, 3\}, \{3, 4\}, \{2, 4\}, \{2, 5\}, \{1, 5\}, \{1, 6\}\}$ (Figure 3.6).

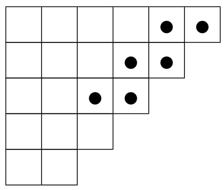


FIGURE 3.6: A portion of the maximal rim hook.

The next subroutine is *isyoungshape* (Figure 3.7), which takes a list of boxes in terms of their positions and test whether they form a Young diagram or not. Note that a collection of boxes form a Young diagram if and only if for every box in the collection, all the boxes to the left and above that box must also belong to the collection.

```
In[23]:= isyoungshape[posns_] :=
       (isin[boxA_] := If[Position[posns, boxA] == {}, False, True];
        testposn[box_] :=
         Apply[And, Join[Map[isin, Table[{k, box[[2]]}, {k, 1, box[[1]], 1}]],
           Map[isin, Table[{box[[1]], k}, {k, 1, box[[2]], 1}]]]];
        Apply[And, Map[testposn, posns]])
```

FIGURE 3.7: The Mathematica code for isyoungshape.

Assume a collection of boxes form a Young diagram, we would like to construct the shape of the Young diagram. Basically, we just need to arrange the positions in lexicographic order, and then take the largest second position among those with the same first position. The Mathematica code for the subroutine *findpartition* is given in Figure 3.8.

```
In[24]:= findpartition[ll_] := (
           takefirst[s_] := s[[1]];
           rows = Union[Map[takefirst, ll]];
           takesecond[t_, s_] := If[ s[[1]] == t , s[[2]] , 1];
           Map[Max , Outer[takesecond , rows , ll , 1]]
          )
```

FIGURE 3.8: The Mathematica code for findpartition.

We move to the μ-broken rim hook decomposition of a partition λ. Before describing the algorithm, let us start with a simple example. Suppose $\lambda = (4,2)$ and $\mu = (2,2,2)$, we proceed as follows.

- Starting with the first element of μ, which is 2, we find all broken rim hooks of λ of length 2. In this case, they are $\{(2,2),(4,2)\}$, $\{(4),(4,2)\}$ and $\{(3,1),(4,2)\}$ (the broken rim hook is obtained by taking the first partition out of the second partition.) We obtain three new partitions $(2,2)$, (4) and $(3,1)$.

- Moving on to the second element of μ, which is 2, we find all broken rim hooks of length 2 for each new partition obtained in the first step. The partition $(2,2)$ has two broken rim hooks $\{(2),(2,2)\}$ and $\{(1,1),(2,2)\}$, the partition (4) has one broken rim hook $\{(2),(4)\}$, the partition $(3,1)$ has two broken rim hooks $\{(1,1),(3,1)\}$ and $\{(2),(3,1)\}$. In this case, we obtain five new partitions.

- For the last element of μ, which is 2, we see that each new partition obtained in step 2 is a broken rim hook of itself. So we obtain five μ-broken rim hook decompositions of λ: $\{(2),(2,2),(4,2)\}$, $\{(1,1),(2,2),(4,2)\}$, $\{(2),(4),(4,2)\}$, $\{(1,1),(3,1),(4,2)\}$ and $\{(2),(3,1),(4,2)\}$.

Figure 3.9 records what we have done graphically.

Basically, given partitions λ and μ, we want to output a "tree" of partitions. This can be achieved by the following two subroutines. The first subroutine is *broken* (Figure 3.10) which takes a sequence of partitions and some integer n, compute all broken rim hooks of length n of the first partition in the sequence and add them to the sequence of partitions. For example, if the sequence of partitions is

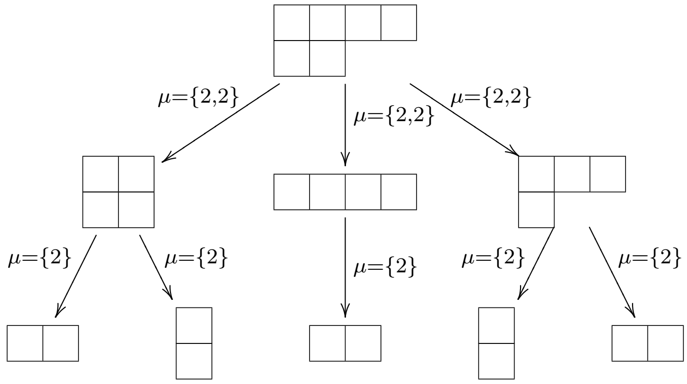


FIGURE 3.9: The μ-broken rim hook decomposition of λ.

$\{(2,2),(4,2)\}$ and $n=2$, then the output will be $\{\{(2),(2,2),(4,2)\},\{(1,1),(2,2),(4,2)\}\}$ (recall that the partition $(2,2)$ has two broken rim hooks of length 2.) The idea for the subroutine is quite simple: for each partition, we construct the maximal rim hook h and find all subsets of h which form a skew shape, i.e., the removal of such a subset gives us a partition. Then we just need to run a loop through all the

```
In[25]:= broken[par_, n_] :=
   Module[{sigma, outcome, a, possiblebrhs, diagsboxes, removeds, R, p},
    sigma = par[[1]]; If[n == 0, outcome = {}, a = rim[sigma, {Length[sigma], 1}];
     possiblebrhs = Subsets[a, {n}];
     diagsboxes =
      Flatten[Table[ Table[{k, l}, {l, 1, sigma[[k]]}], {k, 1, Length[sigma]}],
       1];
     leftover[s_] := Complement[diagsboxes, s];
     removeds = Map[leftover, possiblebrhs];
     evaluate[s_] := If[isyoungshape[s], findpartition[s], R];
     p = Map[evaluate, removeds];
     p = Complement[p, {R}];
     f[s_] := Join[{s}, par];
     outcome = Map[f, p]]; outcome]
```

FIGURE 3.10: The Mathematica code for broken.

elements of μ and apply the subroutine broken to every sequence of partitions in an iteration. To wrap things up, the second subroutine *decompose* (Figure 3.11) will take two partitions λ, μ and output all μ-broken rim hook decompositions of λ. Let's have a look at some more complicated example, suppose $\lambda=(6,5,3,1)$, which has 15 boxes and $\mu=(7,4,3,1)$. The result is illustrated in Figure 3.12.

```
In[26]:= decompose[lambda_, mu_] :=
  Module[{outcome, i}, outcome = broken[{lambda}, mu[[1]]]; i = 2;
   While[i ≤ Length[mu], outcome = Flatten[Map[broken[#, mu[[i]]] &, outcome], 1];
    i++]; outcome]
```

FIGURE 3.11: The Mathematica code for decompose.

```
{{{}, {1}, {4}, {4, 4}, {6, 5, 3, 1}},
 {{}, {1}, {3, 1}, {4, 4}, {6, 5, 3, 1}}, {{}, {1}, {4}, {5, 3}, {6, 5, 3, 1}},
 {{}, {1}, {2, 2}, {5, 3}, {6, 5, 3, 1}}, {{}, {1}, {3, 1}, {5, 3}, {6, 5, 3, 1}},
 {{}, {1}, {4}, {6, 2}, {6, 5, 3, 1}}, {{}, {1}, {2, 2}, {6, 2}, {6, 5, 3, 1}},
 {{}, {1}, {3, 1}, {6, 2}, {6, 5, 3, 1}}, {{}, {1}, {2, 2}, {4, 2, 2}, {6, 5, 3, 1}},
 {{}, {1}, {3, 1}, {4, 2, 2}, {6, 5, 3, 1}},
 {{}, {1}, {2, 1, 1}, {4, 2, 2}, {6, 5, 3, 1}},
 {{}, {1}, {4}, {4, 3, 1}, {6, 5, 3, 1}}, {{}, {1}, {2, 2}, {4, 3, 1}, {6, 5, 3, 1}},
 {{}, {1}, {3, 1}, {4, 3, 1}, {6, 5, 3, 1}},
 {{}, {1}, {2, 1, 1}, {4, 3, 1}, {6, 5, 3, 1}},
 {{}, {1}, {4}, {5, 2, 1}, {6, 5, 3, 1}}, {{}, {1}, {2, 2}, {5, 2, 1}, {6, 5, 3, 1}},
 {{}, {1}, {3, 1}, {5, 2, 1}, {6, 5, 3, 1}},
 {{}, {1}, {2, 1, 1}, {5, 2, 1}, {6, 5, 3, 1}},
 {{}, {1}, {4}, {4, 2, 1, 1}, {6, 5, 3, 1}},
 {{}, {1}, {2, 2}, {4, 2, 1, 1}, {6, 5, 3, 1}},
 {{}, {1}, {3, 1}, {4, 2, 1, 1}, {6, 5, 3, 1}},
 {{}, {1}, {2, 1, 1}, {4, 2, 1, 1}, {6, 5, 3, 1}},
 {{}, {1}, {1, 1, 1, 1}, {4, 2, 1, 1}, {6, 5, 3, 1}}}
```

FIGURE 3.12: All μ-broken rim hook decompositions of λ.

3.5 Computing the Weights

We've tackled the decomposition problem, the next thing is to calculate the weight of a broken rim hook. Essentially, we just need to know the number of rim hooks contained in the broken rim hook and the number of rows and columns of each rim hook. To facilitate the process, we represent each broken rim hook by a list of sequences of 0 and 1: 1 means the box belongs to the broken rim hook and 0 means it doesn't. For instance, the broken rim hook $(6,5,4,3,2)/(4,4,4,2)$ (Figure 3.4) can be represented by $\{(0,0,0,0,1,1),(0,0,0,0,1),(0,0,0,0),(0,0,1),(1,1)\}$. The subroutine *skewdiagram* (Figure 3.13) takes a skew diagram λ/μ and represents it in the above form.

```
skewdiagram[l_, m_] :=
 Module[{s, p, f}, p = Join[m, Table[0, {i, 1, Length[l] - Length[m]}]];
  s = Array[f, Length[l]];
  Do[s[[i]] = Join[Table[0, {i, 1, p[[i]]}], Table[1, {i, 1, l[[i]] - p[[i]]}]],
   {i, 1, Length[l]}]; s]
```

FIGURE 3.13: The Mathematica code for skewdiagram.

Now it is quite easy to read off the number of rim hooks as well as their number of rows and columns. For a broken rim hook λ/μ, first we scan it from top to bottom to find the first row which is not all 0. If there is such a row, i.e., we've found a rim hook, we increase the number of rim hooks by 1 and set the number of columns of that rim hook to be the total number of 1 in that row and the number of rows of that rim hook to be 1. We then compare that row to the next row, if they agree on some nonzero positions, then we increase the number of columns by the total number of 1 in the next row minus 1 and increase the number of rows by 1. We move on to the next row and continue until the condition is no longer true. In that case, we'll search for the next first row which is not all 0 and repeat the same procedure until the last row is reached. The idea is illustrated in the subroutine *weightbrhs* (Figure 3.14), which takes a broken rim hook and outputs the corresponding weight. So for example, if we consider the broken rim

```
weightbrhs[lambda_, mu_] :=
 Module[{r, c, n, w, t, s}, s = skewdiagram[lambda, mu]; t = 1; w = 1;
  n = 0; While[t <= Length[s],
   While[! MemberQ[s[[t]], 1] && t < Length[lambda], t++];
   Which[t < Length[s] || Total[s[[Length[s]]]] ≠ 0, n++; c = Total[s[[t]]];
    r = 1];
   While[t < Length[s] && Position[s[[t]], 1, 1, 1] == {{Length[s[[t + 1]]]}},
    r++; c = c + Total[s[[t + 1]]] - 1; t++];
   Which[t < Length[s] || Total[s[[Length[s]]]] ≠ 0,
    w *= (-1) ^ (r - 1) q^ (2 (c - 1))]; t++]; w *= (q^2 - 1) ^ (n - 1); w]
```

FIGURE 3.14: The Mathematica code for weightbrhs.

hook given in Figure 3.4, then the weight is given by

$$(q^2 - 1)^2(-1)q^2q^2 = -q^4(q^2 - 1)^2.$$

Now we can calculate the weight of a μ-broken rim hook decomposition, which is just a product of the broken rim hooks, and the character formula follows (Figure 3.15 and Figure 3.16).

```
weight[decomp_] :=
 Module[{w = 1}, Do[w *= weightbrhs[decomp[[i + 1]], decomp[[i]]],
   {i, 1, Length[decomp] - 1}]; w]
```

FIGURE 3.15: The Mathematica code for weight.

In Figure 3.17, we provide the table of characters of $H_5(q)$ evaluated on the partitions of 5.

```
character[lambda_, mu_] :=
 Module[{d, result}, d = decompose[lambda, mu];
  result = Simplify[q^(-Total[mu] + Length[mu]) * Apply[Plus, Map[weight, d]]];
  result]
```

FIGURE 3.16: The Mathematica code for character.

$\lambda\backslash\mu$	(1^5)	(21^3)	(2^21)	(31^2)	(32)	(41)	(5)
(1^5)	1	$-q^{-1}$	q^{-2}	q^{-2}	$-q^{-3}$	$-q^{-3}$	q^{-4}
(21^3)	4	$q-3q^{-1}$	$-2+2q^{-2}$	$-1+2q^{-2}$	$2q^{-1}-q^{-3}$	$q^{-1}-q^{-3}$	$-q^{-2}$
(2^21)	5	$2q-3q^{-1}$	q^2-2+2q^{-2}	$-2+q^{-2}$	$-q+q^{-1}-q^{-3}$	q^{-1}	0
(31^2)	6	$3q-3q^{-1}$	q^2-4+q^{-2}	q^2-2+q^{-2}	$-2q+2q^{-1}$	$-q+q^{-1}$	1
(32)	5	$3q-2q^{-1}$	$2q^2-2+q^{-2}$	q^2-2	q^3-q+q^{-1}	$-q$	0
(41)	4	$3q-q^{-1}$	$2q^2-2$	$2q^2-1$	q^3-2q	q^3-q	$-q^2$
(5)	1	q	q^2	q^2	q^3	q^3	q^4

FIGURE 3.17: Characters of $H_5(q)$.

3.6 The Character Decomposition Algorithm

For a positive permutation braid T_σ for $\sigma \in S_n$, i.e., a basis element of $H_n(q)$, it is stated in [11] that there exists a $\mathbb{C}(q)$ linear combination

$$c_\sigma = \sum_{\mu \vdash n} a_{\sigma\mu} T_{\gamma_\mu},$$

$a_{\sigma\mu} \in \mathbb{C}(q)$ such that $\zeta^\lambda(c_\sigma) = \zeta^\lambda(T_\sigma)$ for all partitions λ of n. In other words, the character of any basis element of $H_n(q)$ is completely determined by the characters of these T_{γ_μ}, where μ is a partition of n. We'll describe how to decompose the character of a positive permutation braid into a linear combination of the characters of these special braids.

Firstly it can be shown that if c is a *composition* of n and μ is the partition obtained by rearranging the parts of c in non-increasing order, then $\zeta^\lambda(T_{\gamma_c}) = \zeta^\lambda(T_{\gamma_\mu})$ for all partitions λ of n. Thus it suffices to find a $\mathbb{C}(q)$ linear combination

$$d_\sigma = \sum_{c \models n} b_{\sigma c} T_{\gamma_c},$$

$b_{\sigma c} \in \mathbb{C}(q)$ such that $\zeta^\lambda(T_\sigma) = \zeta^\lambda(d_\sigma)$ for all partition λ of n.

Consider a permutation $\sigma \in S_n$ and let i be the first position such that $\sigma(i) > i+1$. Note that if such i doesn't exist, then the permutation is already of the form

γ_c for some composition c of n and we're done. Otherwise we let $j = \sigma(i) - 1$ and consider two cases:

Case 1 $\sigma(j) < \sigma(j+1)$, then

$$\zeta^\lambda(T_\sigma) = \zeta^\lambda(T_{s_j \sigma s_j}).$$

Note that for the permutation $\sigma' = s_j \sigma s_j$, we have $\sigma'(i) = j = \sigma(i) - 1$.

Case 2 $\sigma(j) > \sigma(j+1)$, then

$$\zeta^\lambda(T_\sigma) = (q-1)\zeta^\lambda(T_{\sigma s_j}) + q\zeta^\lambda(T_{s_j \sigma s_j}).$$

Again for each of the permutations $\sigma' = \sigma s_j$ and $\sigma' = s_j \sigma s_j$ we have $\sigma'(i) = j = \sigma(i) - 1$.

If we continue to apply the procedure, we'll obtain the decomposition of $\zeta^\lambda(T_\sigma)$ after finitely many steps because there are only finitely many strands.

Let us take an example now. For simplicity, we'll omit writing the character and just consider permutations instead of positive permutation braids. Suppose we are given the permutation $(3,4,2,1)$ (recall that it means $\sigma(1) = 3$, $\sigma(2) = 4$, $\sigma(3) = 2$, $\sigma(4) = 1$), again it is helpful to express the decomposition in term of tree diagrams (Figure 3.18). We see immediately that the character of $T_{(3421)}$ is

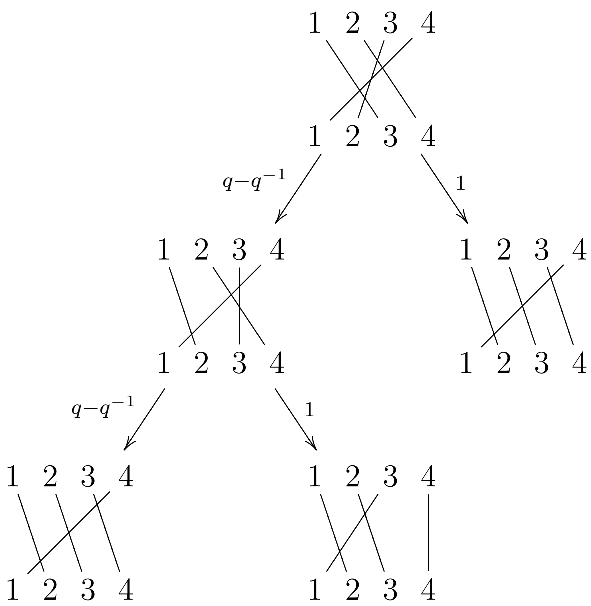


FIGURE 3.18: The character decomposition.

given by

$$\begin{aligned}\zeta^\lambda(T_{(3421)}) &= [(q-q^{-1})^2+1]\zeta^\lambda(T_{(2341)}) + (q-q^{-1})\zeta^\lambda(T_{(2314)}) \\ &= (q^2-1+q^{-2})\zeta^\lambda(T_{\gamma_4}) + (q-q^{-1})\zeta^\lambda(T_{\gamma_3\times\gamma_1}).\end{aligned}$$

To implement the character decomposition procedure in Mathematica, we'll employ several subroutines. The subroutine *permute* (Figure 3.19) will take a permutation in S_n and a list of n elements and output the permuted list under the action of the permutation. Note that if $\sigma \in S_n$, then

$$\sigma(\{x_1, x_2, ..., x_n\}) = \{x_{\sigma(1)}, x_{\sigma(2)}, ..., x_{\sigma(n)}\}$$

and we represent the permutation σ as a list $\{\sigma(1), \sigma(2), ..., \sigma(n)\}$. The subroutine *permproduct* (Figure 3.19) will output the product of two permutations σ and τ in S_n.

```
In[32]:= permute[perm_, list_] := Module[{p}, result[i_] := list[[perm[[i]]]];
           p = Map[result, Table[i, {i, 1, Length[list]}]]; p]

In[33]:= permproduct[perm1_, perm2_] :=
          Module[{p, l}, l = Table[i, {i, 1, Length[perm1]}]; p = permute[perm2, l];
           p = permute[perm1, p]; p]
```

FIGURE 3.19: The subroutines permute and permproduct.

The next subroutine is called *cross* (Figure 3.20), which takes a permutation σ and outputs the first position i where $\sigma(i) > i+1$. If no such position is found, the output will be 0.

```
In[34]:= cross[perm_] := Module[{i, n, output},
           i = 1; n = Length[perm];
           While[perm[[i]] ≤ i + 1 && i < n, i++];
           If[i == n, output = 0, output = i]; output]
```

FIGURE 3.20: The subroutine cross.

Assume we've found the first position i where $\sigma(i) > i+1$, the subroutine *replace* (Figure 3.22) will take a list consisting of the permutation σ and i, apply the decomposition algorithm we described and output the result according to case 1 or case 2. The output will be a list where each element is a list consisting of a permutation together with its coefficient. Note that the subroutine *permrepr* (Figure 3.21) takes a braid b on n strands and outputs its corresponding permutation.

```
In[6]:= permrepr[n_, b_] := Module[{p, t}, p = Table[i, {i, 1, n}];
          Do[t = p[[Abs[b[[i]]]]]; p[[Abs[b[[i]]]]] = p[[Abs[b[[i]]] + 1]];
           p[[Abs[b[[i]]] + 1]] = t, {i, Length[b], 1, -1}]; p]
```

FIGURE 3.21: The subroutine permrepr.

input={permutaion, coef}

```
In[52]:= replace[input_, i_] := Module[{perm, coef, j, sj, n, output},
          perm = input[[1]]; coef = input[[2]]; n = Length[perm]; j = perm[[i]] - 1;
          sj = permrepr[n, {j}]; Which[perm[[j]] < perm[[j + 1]],
           output = {{permproduct[permproduct[sj, perm], sj], coef}},
           perm[[j]] > perm[[j + 1]],
           output = {{permproduct[perm, sj], coef * (q - q^-1)},
             {permproduct[permproduct[sj, perm], sj], coef}}]; output]
```

FIGURE 3.22: The subroutine replace.

The next subroutine is *mainreplace* (Figure 3.23). Its input is of the form

$$\text{input} = \{\{\{\text{perm},\ \text{coef}\}, \{\text{perm},\ \text{coef}\}, ..., \{\text{perm},\ \text{coef}\}\}, \text{testcross}\}.$$

So an input is a list consisting of two sublists. The first sublist represents a linear combination of permutations over $\mathbb{C}(q)$ and the second sublist testcross results from applying the subroutine cross to each permutation. So basically we input one level of the decomposition tree (figure 3.18) and output the next level. Thus the output consists of a linear combination of permutations over $\mathbb{C}(q)$ together with its corresponding testcross. Of course we only apply the subroutine replace to the permutations whose corresponding elements in testcross are nonzero.

input={{{perm, coef},{perm, coef},...,{perm, coef}},testbelow}

```
In[54]:= mainreplace[input_] := Module[{r, p, test, perm, zero, nonzero, output, s},
          perm = input[[1]]; test = input[[2]]; p = Position[test, 0];
          zero = Flatten[p, 1];
          nonzero = Complement[Table[i, {i, 1, Length[test]}], zero];
          r = perm[[zero]]; testcross[list_] := Map[cross, list];
          output =
           Flatten[MapThread[replace[#1, #2] &, {Delete[perm, p], test[[nonzero]]}],
            1]; s = testcross[Map[First, output]]; output = Join[r, output];
          s = Join[Table[0, {i, 1, Length[r]}], s]; {output, s}]
```

FIGURE 3.23: The subroutine mainreplace.

Finally, the subroutine *tracedecomp* (Figure 3.24) will take a list representing a linear combination of permutations over $\mathbb{C}(q)$ and output its decomposition. The idea is to apply the subroutine mainreplace repeatedly to the input until testcross consists of all 0.

```
list_={{permutation, coefficient},{permutation, coefficient},...,{permutation, coefficient}}

In[62]:= tracedecomp[list_] :=
  Module[{outcome, test}, test = Map[cross, Map[First, list]];
   outcome = NestWhile[mainreplace, {list, test}, Total[#[[2]]] ≠ 0 &];
   outcome = outcome[[1]]; outcome]
```

FIGURE 3.24: The subroutine tracedecomp.

Let's try some example, suppose we are given the permutation $(4, 6, 2, 1, 3, 5)$, the output is given in figure 3.25. Thus we obtain

```
In[65]:= tracedecomp[{{{4, 6, 2, 1, 3, 5}, 1}}]
```

$$\left\{\left\{\{2,3,4,5,6,1\}, -\frac{1}{q}+q\right\},\ \{\{2,1,4,5,6,3\},1\},\ \left\{\{2,3,4,5,6,1\}, \left(-\frac{1}{q}+q\right)^3\right\},\ \left\{\{2,3,4,5,1,6\}, \left(-\frac{1}{q}+q\right)^2\right\},\ \left\{\{2,3,4,5,6,1\}, -\frac{1}{q}+q\right\}\right\}$$

FIGURE 3.25: The character decomposition of $(4, 6, 2, 1, 3, 5)$.

$$\zeta^\lambda(T_{(462135)}) = \left(q^3 - q - \frac{1}{q^3} + \frac{1}{q}\right)\zeta^\lambda(T_{\gamma_6}) + \zeta^\lambda(T_{\gamma_4\times\gamma_2}) + \left(q^2 - 2 + \frac{1}{q^2}\right)\zeta^\lambda(T_{\gamma_5\times\gamma_1}).$$

3.7 Permutation Braid Decomposition

Ultimately, we want a program which can calculate the character of an arbitrary element of $H_n(q)$ indexed by a partition λ. We've already had most of the ingredients. The remaining task is to express an arbitrary element of $H_n(q)$ as a linear combination of the basis elements, i.e., positive permutation braids. Note that one property of a positive permutation braid is that its ith strand always goes above all the strands to its left. Hence our strategy is as follows.

- Given an arbitrary braid on n strands, we'll make the ith strand go above all the strands to its left. We'll start from the nth strand of the braid and

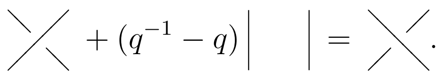


FIGURE 3.26: The skein relation.

go successively to the left. Everytime we see the nth strand go below some other strand, we'll "lift" it up using the *skein* relation

This is just the third relation of the Hecke algebra $H_n(q)$ where each g_i is replaced by the positive crossing. Note that everytime we apply the relation, we'll obtain one more braid with one fewer crossing and since the number of crossings is finite, that guarantees the algorithm will terminate.

- Eventually, we'll obtain a linear combination of braids, where each one has the property that the ith strand goes above all the other strand to its left. Each of these braids is not of the form of a positive permutation braid yet, but it's clear that it is equivalent (isotopic) to a positive permutation braid. Thus we can just replace each one of them by their corresponding permutations.

Let's implement the above idea in Mathematica. To make things easy to follow, we'll break the algorithm into several subroutines. The first subroutine is *skein* (Figure 3.27) which takes two arguments: one is a list consisting of a braid together with its coefficient and a position i in the braid word where we want to apply the skein relation. Note that we represent a braid by its braid word and input to Mathematica as a list. For instance, the braid $\sigma_1\sigma_2^{-1}\sigma_4$ is input as $\{1, -2, 4\}$. The output of course are the braids we obtain after applying the skein relation together with their coefficients.

input={braid,coef}

```
In[46]:= skein[input_, i_] := Module[{b, c, outcome}, b = input[[1]]; c = input[[2]];
  Which[b[[i]] > 0,
   outcome = {{MapAt[Times[-1, #] &, b, i], c}, {Delete[b, i], c * (q - q^-1)}},
   b[[i]] < 0,
   outcome = {{MapAt[Times[-1, #] &, b, i], c}, {Delete[b, i], c * (q^-1 - q)}}];
  outcome]
```

FIGURE 3.27: The subroutine skein.

The next subroutine is *below* (Figure 3.30), which will take a braid b on n strands and test whether the mth strand goes below some other strand to its left. Note that there are two instances where we have to test.

If the mth strand goes above all the other strands to its left, the output will be 0. Otherwise, the subroutine will output the first position in the braid word

FIGURE 3.28: Possibilities for the crossing.

where the mth strand goes below some other strand to its left. Note that the subroutine *permutation* (Figure 3.29) takes a list p of n elements and a generator s_i and ouputs the list obtained by swopping the ith and $i+1$st elements of p. This subroutine is used to keep track of the mth strand.

```
In[5]:= permutation[i_, p_] := Module[{q, j}, j = Abs[i]; q = p; q[[j]] = p[[j + 1]];
          q[[j + 1]] = p[[j]]; q]
```

FIGURE 3.29: The subroutine permutation.

```
In[9]:= below[n_, b_, m_] := Module[{p, outcome}, p = Table[i, {i, 1, n}];
          Do[p = permutation[b[[i]], p];
           If[b[[i]] > 0 && p[[b[[i]] + 1]] == m && p[[b[[i]]]] < m ||
             b[[i]] < 0 && p[[-b[[i]]]] == m && p[[-b[[i]] + 1]] < m, outcome = i;
            Break[], Which[i == Length[b], outcome = 0]], {i, 1, Length[b]}]; outcome]
```

FIGURE 3.30: The subroutine below.

The subroutine *testbelow* (Figure 3.31) takes a list of braids and on n strands and test whether the m strand in each braid goes above all the other strands to its left.

```
In[10]:= testbelow[n_, list_, m_] := Map[below[n, #, m] &, list];
```

FIGURE 3.31: The subroutine testbelow.

So given an element of the Hecke algebra, which is a linear combination of braids, we can apply the subroutine testbelow to test the status of the mth strand in each braid. The subroutine *above* (Figure 3.32) will take a list of braids and perform the skein relation to those whose corresponding elements in testbelow are nonzero. Its input is of the form

$$\text{input} = \{\{\{\text{braid},\ \text{coef}\}, \{\text{braid},\ \text{coef}\}, ..., \{\text{braid},\ \text{coef}\}\}, \text{testbelow}, n, m\}$$

where n is the number of strands of the braids and m is the strand that we want to test. The output will be of the same form as the input.

input={{{braid,coef},{braid,coef},...,{braid,coef}}, testbelow, n, m}

```
In[11]:= above[input_] := Module[{b, test, p, zero, nonzero, r, s, output, n, m},
   b = input[[1]]; test = input[[2]]; n = input[[3]]; m = input[[4]];
   p = Position[test, 0]; zero = Flatten[p, 1];
   nonzero = Complement[Table[i, {i, 1, Length[test]}], zero]; r = b[[zero]];
   output = Flatten[MapThread[skein[#1, #2] &, {Delete[b, p], test[[nonzero]]}],
     1]; s = testbelow[n, Map[First, output], m]; output = Join[r, output];
   s = Join[Table[0, {i, 1, Length[r]}], s]; {output, s, n, m}]
```

FIGURE 3.32: The subroutine above.

Having built the subroutine above, now we just need to apply it repeatedly to a list of braids so long as testbelow is not all zero. The result that we obtain will be a list of braids where each of them has the mth strand go above all the strands to its left. The subroutine *mainabove* (Figure 3.33) takes as its input:

$$\text{input} = \{\{\{\text{braid},\ \text{coef}\}, \{\text{braid},\ \text{coef}\}, ..., \{\text{braid},\ \text{coef}\}\}, n, m\}$$

where n is the number of strands of the braids and m is the strand that we want to test.

input={{{braid,coef},{braid,coef},...,{braid,coef}}, n, m}

```
In[12]:= mainabove[input_] := Module[{test, b, n, m, output, list}, list = input[[1]];
   b = Map[First, list]; n = input[[2]]; m = input[[3]]; test = testbelow[n, b, m];
   output = NestWhile[above, {list, test, n, m}, Total[#[[2]]] ≠ 0 &];
   output = output[[1]]; {output, n, m - 1}]
```

FIGURE 3.33: The subroutine mainabove.

Thus given an arbitrary braid, we can apply the subroutine mainabove to obtain the list of braids where each of them has the mth strand go above all the strands to its left. To obtain all permutation braids, we just need to apply mainabove repeatedly starting from the nth strand and go successively to the left until the first strand is reached. Eventually we obtain a linear combination of braids where each one of them is equivalent to a positive permutation braid. To retrieve the permutation braid, we just need to apply the subroutine permrepr to each braid in the result. The subroutine *permbraiddecomp* (Figure 3.34) takes a braid on n strands and expresses it in terms of positive permutation braids.

Let's try an example. Suppose we are given the braid $\{-4, -3, -3, 2, -4, 3, -2\}$ on 5 strands. Its permutation braid decomposition is given in Figure 3.35. So we obtain 17 permutation braids (some of them may represent the same permutation though). Depending on the number of strands and the nature of the crossings, input={braid,coef}

```
In[13]:= permbraiddecomp[n_, input_] :=
  Module[{outcome}, outcome = NestWhile[mainabove, {{input}, n, n}, #[[3]] > 1 &];
   outcome = outcome[[1]]; outcome]
```

FIGURE 3.34: The subroutine permbraiddecomp.

```
In[53]:= permbraiddecomp[5, {{-4, -3, -3, 2, -4, 3, -2}, 1}]

Out[53]=
```

$$\left\{\{\{4,3,-3,2,-4,3,2\},1\},\left\{\{4,3,-3,2,-4,3\},\frac{1}{q}-q\right\},\left\{\{4,3,2,4,3,-2\},\frac{1}{q}-q\right\},\left\{\{4,3,2,3,-2\},\left(\frac{1}{q}-q\right)^2\right\},\left\{\{4,2,-4,3,2\},\left(\frac{1}{q}-q\right)^2\right\},\left\{\{4,2,-4,3\},\left(\frac{1}{q}-q\right)^3\right\},\left\{\{3,-3,2,4,3,2\},\frac{1}{q}-q\right\},\left\{\{3,-3,2,4,3\},\left(\frac{1}{q}-q\right)^2\right\},\left\{\{3,-3,2,3,2\},\left(\frac{1}{q}-q\right)^2\right\},\left\{\{3,-3,2,3\},\left(\frac{1}{q}-q\right)^3\right\},\left\{\{3,2,3,-2\},\left(\frac{1}{q}-q\right)^3\right\},\left\{\{3,2,4,3,2\},\left(\frac{1}{q}-q\right)^2\right\},\left\{\{2,4,3,2\},\left(\frac{1}{q}-q\right)^3\right\},\left\{\{3,2,4,3\},\left(\frac{1}{q}-q\right)^3\right\},\left\{\{2,4,3\},\left(\frac{1}{q}-q\right)^4\right\},\left\{\{2,3,2\},\left(\frac{1}{q}-q\right)^4\right\},\left\{\{2,3\},\left(\frac{1}{q}-q\right)^5\right\}\right\}$$

FIGURE 3.35: The permutation decomposition of $\{-4,-3,-3,2,-4,3,-2\}$.

we get different number of permutation braids. For instance the following braid $\{4,3,5,4,-2,-3,-1,-2,4,3,5,4,-2,-3,-1,-2\}$ on 6 strands has 584 terms in the result.

We still need one small subroutine. The subroutine *part* (Figure 3.36) takes a permutation γ_c and outputs its cycle type, arranged in non-increasing order. For instance, the permutation $(1,2,4,3)$ has cycle type $(2,1,1)$. The idea is quite simple: we just need to go through γ_c. Everytime we find $\gamma_c(i) < i$, we add $i+1-\gamma_c(i)$ to the result and everytime we find $\gamma_c(i) = i$, we add 1 to the result.

```
In[38]:= part[p_] := Module[{b = {}, f = p},
   Do[Which[f[[i]] < i, b = Append[b, -f[[i]] + i + 1], f[[i]] == i,
     b = Append[b, 1]], {i, 1, Length[f]}]; b = Sort[b, Greater]; b]
```

FIGURE 3.36: The subroutine part.

3.8 The Main Program

We can now put everything together in the subroutine *maincharacter* (Figure 3.37), which takes a partition λ of n, a linear combination of braids on n strands and output the character ζ^λ. Given a braid b on n strands and a partition λ of n, we summarize the steps to calculate $\zeta^\lambda(b)$:

- express b as a linear combination of positive permutation braids T_σ,
- express each permutation braid in the decomposition of b as linear combination of T_{γ_μ},
- employ the quantized Murnaghan-Nakayama rule to calculate $\zeta^\lambda(\mu)$.

```
input={{braid, coef}, {braid,coef},..., {braid,coef}}

maincharacter[lambda_, input_] :=
 Module[{n, a, outcome}, n = Total[lambda];
  a = Flatten[Map[permbraiddecomp[n, #] &, input], 1];
  replace1[a_, i_] := {permrepr[n, a[[i]][[1]]], a[[i]][[2]]};
  a = Map[replace1[a, #] &, Table[i, {i, 1, Length[a]}]];
  replace2[a_, i_] := tracedecomp[{a[[i]]}];
  a = Flatten[Map[replace2[a, #] &, Table[i, {i, 1, Length[a]}]], 1];
  replace3[a_, i_] := {part[a[[i]][[1]]], a[[i]][[2]]};
  a = Map[replace3[a, #] &, Table[i, {i, 1, Length[a]}]];
  replace4[a_, i_, l_] := a[[i]][[2]] * character[lambda, a[[i]][[1]]];
  a = Map[replace4[a, #, lambda] &, Table[i, {i, 1, Length[a]}]];
  outcome = Simplify[Total[a]]; outcome]
```

FIGURE 3.37: The subroutine maincharacter.

Chapter 4

Link Invariants from Quantum Groups

4.1 Representations of the Artin Braid Groups

An n-*braid* (Figure 4.1) is a set of n non-intersecting strands, all of which are attached to a horizontal bar at the top and at the bottom. Each strand always heads downward as we move along any one of the strands from the top bar to the bottom bar. In other words, each string intersects any horizontal plane between the two bars exactly once. We consider two braids to be the same (equivalent) if we can rearrange the strings in the two braids to look the same without passing any strings through one another or themselves while keeping the endpoints fixed.

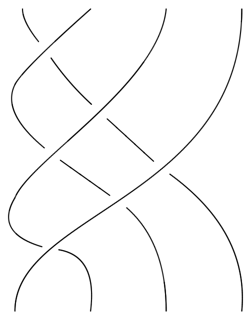


FIGURE 4.1: A braid on 4 strands.

Two n-braids can be multiplied by putting one on top of the other. The *identity braid* is the braid where all the strands go straight from top to bottom. It is easy to check that the multiplication of a braid with the identity braid gives us back the same braid and the multiplication of any braid with its reflection over the horizontal plane gives us the identity braid. Thus the set of n-braids form a group under multiplication, which we denote by B_n.

In 1924, the mathematician Emil Artin gave a presentation of the braid group B_n. Specifically, the braid group B_n on n strands can be defined as the group generated by $\sigma_1, ..., \sigma_{n-1}$ and the relations (Figures 4.3 and 4.4)

$$\begin{cases} \sigma_i\sigma_j = \sigma_j\sigma_i, & |i-j| > 1, \\ \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j, & |i-j| = 1. \end{cases}$$

The generator σ_i can be visualized as the braid where all the strands go straight, except for the i^{th} and $i+1^{\text{st}}$ strands (Figure 4.2).

Figure 4.2: The generators σ_i and σ_i^{-1}.

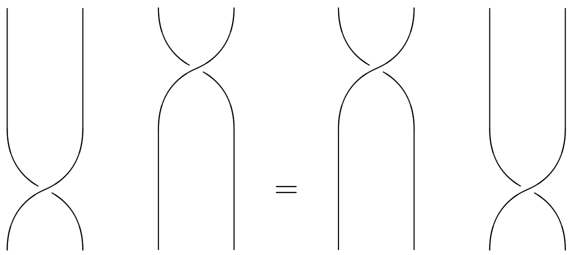


Figure 4.3: The relation $\sigma_i\sigma_j = \sigma_j\sigma_i$.

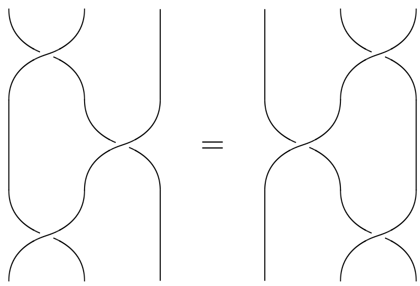


Figure 4.4: The relation $\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}$.

Let V be a complex finite-dimensional vector space and $c\colon V^{\otimes 2} \to V^{\otimes 2}$ is an invertible linear transformation. For $1 \leq i \leq n-1$, we define $c_i \in \mathrm{Aut}(V^{\otimes n})$ by

$$c_i = \begin{cases} c \otimes \mathrm{id}_{V^{\otimes(n-2)}}, & i = 1, \\ \mathrm{id}_{V^{\otimes(i-1)}} \otimes c \otimes \mathrm{id}_{V^{\otimes(n-i-1)}}, & 1 < i < n-1, \\ \mathrm{id}_{V^{\otimes(n-2)}} \otimes c, & i = n-1. \end{cases}$$

We consider the map $h_V \colon B_n \to \mathrm{Aut}(V^{\otimes n})$ given by $\sigma_i \mapsto c_i$. To obtain a representation of B_n, these c_i have to satisfy

$$\begin{cases} c_i c_j = c_j c_i, & |i-j| > 1, \\ c_i c_j c_i = c_j c_i c_j, & |i-j| = 1. \end{cases}$$

The first relation is clearly satisfied. For the second relation, we require that

$$(c \otimes \mathrm{id}_V)(\mathrm{id}_V \otimes c)(c \otimes \mathrm{id}_V) = (\mathrm{id}_V \otimes c)(c \otimes \mathrm{id}_V)(\mathrm{id}_V \otimes c).$$

This equation is called the *Yang-Baxter equation*, which plays an important role in statistical mechanics. Every automorphism of $V^{\otimes 2}$ which satisfies the Yang-Baxter equation is called an *R-matrix*. For example, suppose V has basis $\{v_1, ..., v_N\}$, it can be checked that the following linear operator

$$c(v_i \otimes v_j) = \begin{cases} q v_i \otimes v_i, & i = j, \\ v_j \otimes v_i, & i < j, \\ v_j \otimes v_i + (q - q^{-1}) v_i \otimes v_j, & i > j, \end{cases}$$

where q is a nonzero complex parameter, is an R-matrix.

The connection between braids and links is as follows: given a braid β, we can obtain a link L by closing up the corresponding endpoints of β (Figure 4.5). Conversely, it was proved by J.W. Alexander in 1923 that every oriented link has a closed braid representative. Furthermore, Markov's theorem tells us that the closures of two braids represent the same oriented link if and only if they are related through a sequence of moves in the braid group together with two additional moves, called the Markov moves: $\beta \leftrightarrow \sigma_i \beta \sigma_i^{-1}$ for $i = 1, ..., n-1$ (Figure 4.6) and $\beta \leftrightarrow \beta \sigma_n^{\pm 1}$ (Figure 4.7), where $\beta \in B_n$ (in the second Markov move, we consider β as a braid in B_{n+1}, with the last strand go straight across).

4.2 Quantum Groups

From the connection between braids and oriented links, one can obtain a link invariant from a collection of R-matrix (representation of the braid group) together with some special elements. Right now, there is a procedure to produce these ingredients from a mathematical "machine" called *quantum group*. Interested readers can look in [14] for the full details. In this section, we only sketch

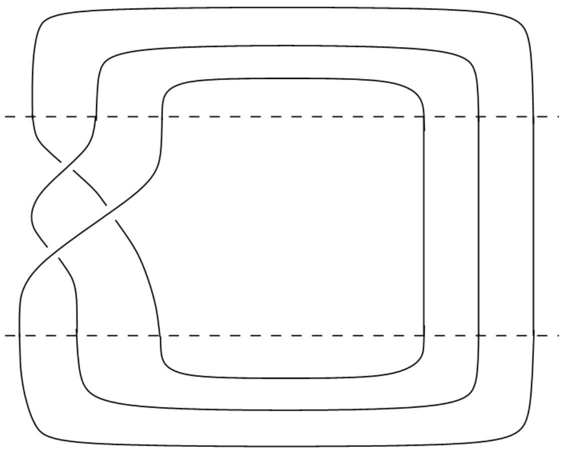


FIGURE 4.5: The closure of $\sigma_1\sigma_2\sigma_1$.

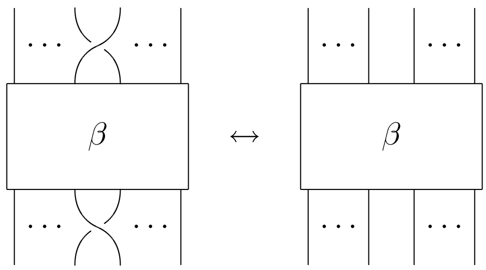


FIGURE 4.6: The first Markov move.

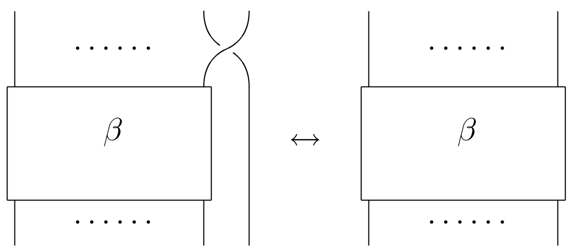


FIGURE 4.7: The second Markov move.

the general picture. In the next section, we shall extract some of the ideas to define the colored HOMFLY polynomial.

To start off, we need to enlarge the family of braids to the family of *oriented tangle*, where we allow the strands to go up and the existence of some knotted loops. Thus an oriented link is a special kind of tangle which doesn't intersect the top and bottom bars. Again two tangles are considered to be the same if there is a continuous deformation of one into the other fixing the endpoints. The composition of two tangles is obtained by putting one on top of the other and hence only makes sense if the ending of one is compatible with the beginning of the other, i.e., the number of points and directions of the arrows are the same. The tensor product of two tangles is obtained by simply putting one next to the other. In the language of category theory, the isotopy classes of tangles form the morphisms between nonnegative integers.

It turns out that the family of tangles can be generated by a finite collection of elementary tangles, subject to a finite set of relations. The next step is to construct some algebraic catogory that admits a functor from the category of tangles (hence we obtain a tangle invariant, and in particular, a link invariant). A natural algebraic category to look for is the category of representations of some algebra (with morphisms being homomorphisms between representations), which leads to the notion of quantum groups.

Despite its name, quantum groups are actually algebras which satisfy a list of axioms. These axioms are designed in such a way that the representations that we obtain from these objects give us our desired algebraic category. For instance, these axioms allow us to take tensor product of two representations and the dual of a representation. They also encode an R-matrix and a bunch of special elements in the algebra. There is a long list of axioms that a quantum group must satisfy and we shall not describe them here. A very good account of quantum groups can be found in [15].

There is a general procedure to construct quantum groups starting from simple Lie algebras via the quantized universal enveloping algebra. In this thesis, we shall utilize the quantum group obtained from the simple Lie algebra sl_N, the Lie algebra of $N \times N$ matrices with zero trace. First, let's recall the presentation of the classical Lie algebra sl_N. There are three special types of generators

$$E_i, \quad F_i, \quad H_i, \ i = 1, .., N-1,$$

where E_i can be thought of as an $N \times N$ matrix with a 1 at position $(i, i+1)$ and zero everywhere else, F_i can be thought of as an $N \times N$ matrix with a 1 at position $(i+1, i)$ and zero everywhere else, H_i can be thought of as an $N \times N$ matrix with a 1 at position (i, i), a -1 at position $(i+1, i+1)$ and zero everywhere else. The relations are

1. $[H_i, H_j] = 0.$
2. $[H_i, E_j] = \alpha_{ji} E_j, \quad [H_i, F_j] = -\alpha_{ji} F_j.$
3. $[E_i, F_j] = \delta_{ij} H_i.$
4. $\begin{cases} [E_i, E_j] = 0, & |i-j| > 1, \\ [E_i, [E_i, E_j]] = 0, & |i-j| = 1. \end{cases}$

5. $\begin{cases} [F_i, F_j] = 0, & |i-j| > 1, \\ [F_i, [F_i, F_j]] = 0, & |i-j| = 1. \end{cases}$

The integers α_{ji} are obtained by calculating the brackets when viewing the generators as matrices. Let q be a nonzero complex number not a root of unity, the *quantized universal enveloping algebra* $U_q(sl_N)$ is a complex algebra generated by

$$E_i, \quad F_i, \quad K_i^{\pm 1}, \ i = 1, ..., N-1$$

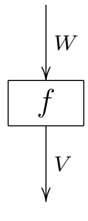
and relations

1. $K_i K_i^{-1} = K_i^{-1} K_i = 1.$
2. $K_i K_j = K_j K_i.$
3. $K_i E_j K_i^{-1} = q^{\alpha_{ji}} E_j, \quad K_i F_j K_i^{-1} = q^{-\alpha_{ji}} F_j.$
4. $E_i F_j - F_j E_i = \delta_{ij} \frac{K_i - K_i^{-1}}{q - q^{-1}}.$
5. $\begin{cases} E_i E_j = E_j E_i, & |i-j| > 1, \\ E_i^2 E_j - (q+q^{-1}) E_i E_j E_i + E_j E_i^2 = 1, & |i-j| = 1. \end{cases}$
6. $\begin{cases} F_i F_j = F_j F_i, & |i-j| > 1, \\ F_i^2 F_j - (q+q^{-1}) F_i F_j F_i + F_j F_i^2 = 1, & |i-j| = 1. \end{cases}$

The integers α_{ji} are the same as in the classical case.

4.3 Graphical Calculus

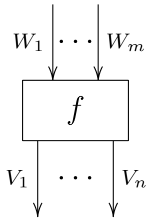
In this section, we introduce some notations which will be useful in the next section. Let V and W be vector spaces, a linear transformation $f\colon V \to W$ can be presented graphically as



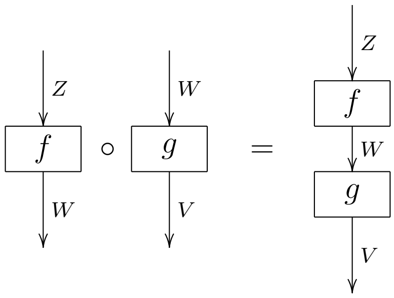
More generally, we can present a linear transformation

$$f\colon V_1 \otimes \cdots \otimes V_n \to W_1 \otimes \cdots \otimes W_m$$

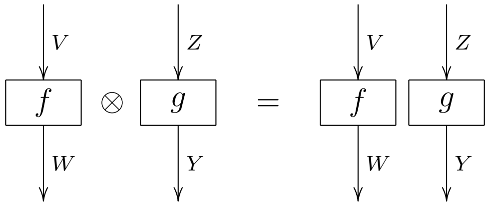
as



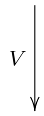
The composition $f \circ g$ of two linear transformations $f\colon W \to Z$ and $g\colon V \to W$ is obtained by putting f on top of g



The tensor product $f \otimes g$ of two linear transformations $f\colon V \to W$ and $g\colon Y \to Z$ is obtained by putting f next to g



We present the identity map $\mathrm{id}\colon V \to V$ by



Let V and W be finite-dimensional vector spaces and $f\colon V \otimes W \to V \otimes W$ is a linear transformation. Suppose V has basis $\{v_1, ..., v_n\}$ and W has basis $\{w_1, ..., w_m\}$ and

$$f(v_i \otimes w_j) = \sum_{\substack{1 \le k \le n \\ 1 \le l \le m}} a_{i,j}^{k,l} v_k \otimes w_l.$$

The usual trace of f is given by

$$\mathrm{tr}(f) = \sum_{\substack{1 \le i \le n \\ 1 \le j \le m}} a_{i,j}^{i,j}.$$

The *partial trace* of f, $\mathrm{tr}'(f)$, is a linear transformation $V \to V$ given by

$$\mathrm{tr}'(f)(v_i) = \sum_{\substack{1 \le k \le n \\ 1 \le j \le m}} a_{i,j}^{k,j} v_k.$$

It follows from linear algebra that

$$\mathrm{tr}(\mathrm{tr}'(f)) = \mathrm{tr}(f).$$

4.4 The Colored HOMFLY Polynomial

Consider the quantum group $U_q(sl_N)$ where q is a nonzero complex number not a root of unity, the class of finite-dimensional representations together with homomorphisms between representations form a *ribbon category* under the usual tensor product. That is, we have the following objects.

Associated to each pair of $U_q(sl_N)$-modules V, W is a natural isomorphism (R-matrix) $R_{V,W} \colon V \otimes W \to W \otimes V$ (Figure 4.8). The naturality (Figure 4.9) means

$$(f \otimes g)R_{Z,W} = R_{Y,V}(g \otimes f),$$

where $f \in \mathrm{Hom}_{U_q(sl_N)}(W, V)$ and $g \in \mathrm{Hom}_{U_q(sl_N)}(Z, Y)$. That is, the following diagram commutes

$$\begin{tikzcd}
Y \otimes V \arrow[r, "R_{Y,V}"] & V \otimes Y \\
Z \otimes W \arrow[u, "g \otimes f"] \arrow[r, "R_{Z,W}"'] & W \otimes Z \arrow[u, "f \otimes g"']
\end{tikzcd}$$

These R-matrices satisfy the triangle equalities (Figure 4.10 and 4.11).

$$R_{U \otimes V, W} = (R_{U,W} \otimes \mathrm{id}_V)(\mathrm{id}_U \otimes R_{V,W}),$$

$$R_{U, V \otimes W} = (\mathrm{id}_V \otimes R_{U,W})(R_{U,V} \otimes \mathrm{id}_W).$$

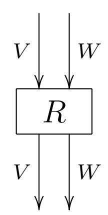


FIGURE 4.8: The isomorphism $R_{V,W}$.

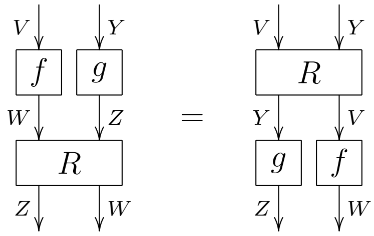


FIGURE 4.9: The naturality relation.

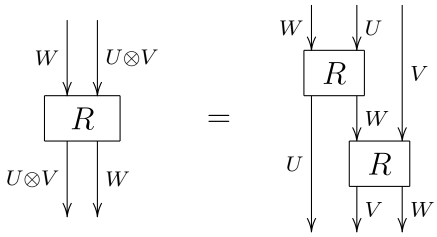


FIGURE 4.10: The first triangle equality.

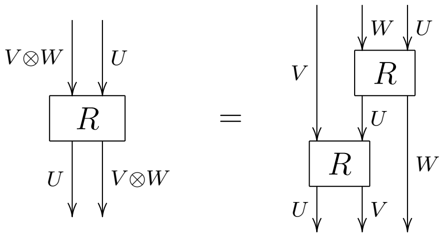


FIGURE 4.11: The second triangle equality.

With these properties, we show that these R-matrices indeed satisfy the Yang-Baxter equation (Figure 4.12), that is

$$
\begin{aligned}
&(R_{V,W} \otimes \mathrm{id}_U)(\mathrm{id}_V \otimes R_{U,W})(R_{U,V} \otimes \mathrm{id}_W) \\
&\qquad\qquad\qquad = (\mathrm{id}_W \otimes R_{U,V})(R_{U,W} \otimes \mathrm{id}_V)(\mathrm{id}_U \otimes R_{V,W}).
\end{aligned}
$$

It is instructive to prove in terms of diagrams, as we illustrate in Figure 4.13,

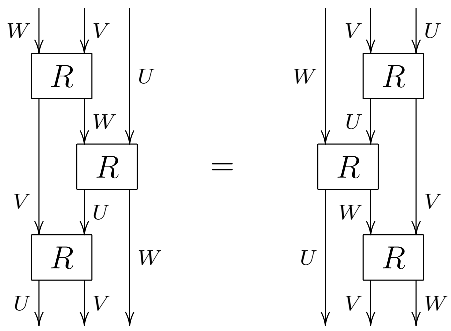


FIGURE 4.12: The Yang-Baxter equation.

where the first and third equalities come from the first triangle equality (Figure 4.10) and the second equality comes from the naturality of R-matrices (Figure 4.9).

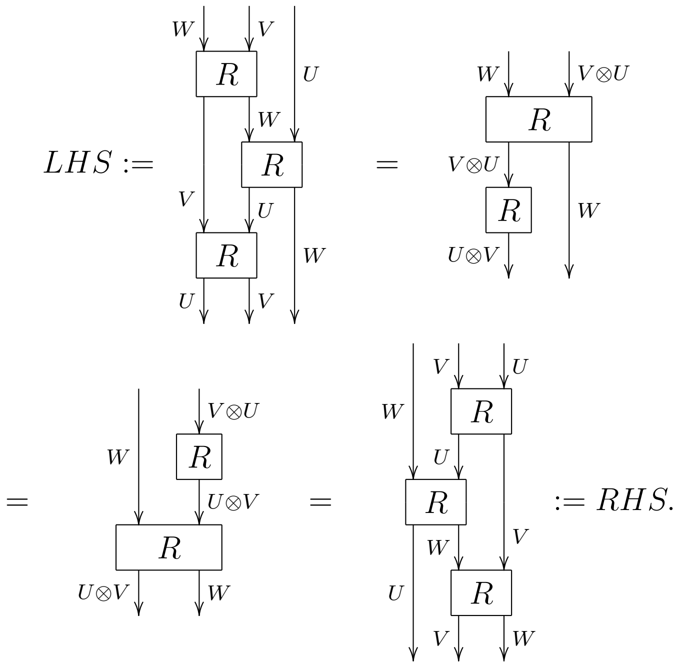


FIGURE 4.13: Proof of the Yang-Baxter equation.

There exists an element $K \in U_q(sl_N)$ such that for any $U_q(sl_N)$-modules V, W we have

$$K \cdot (v \otimes w) = K \cdot v \otimes K \cdot w$$

for $v \in V$ and $w \in W$. For any module homomorphism $f\colon V \otimes W \to V \otimes W$, the *partial quantum trace* (Figure 4.14), tr'_q, of f is defined as

$$\mathrm{tr}'_q(f) = \mathrm{tr}'[(\mathrm{id}_V \otimes K) \circ f].$$

The *quantum trace* (Figure 4.15), tr_q, of f is

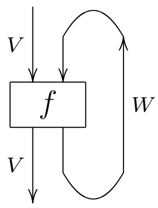


FIGURE 4.14: The partial quantum trace of f.

$$\mathrm{tr}_q(f) = \mathrm{tr}[(K \otimes K) \circ f].$$

The partial quantum trace has the following properties, which follow from linear

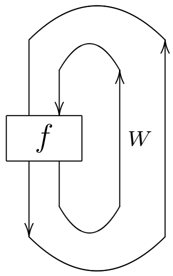


FIGURE 4.15: The quantum trace of f.

algebra.

(i) For morphism $f\colon V \otimes W \to V \otimes W$ and $g\colon V \to V$, we have (Figure 4.16)

$$\mathrm{tr}'_q(f \circ g) = \mathrm{tr}'_q(f) \circ g.$$

(ii) For any morphism $f\colon V \otimes W \to V \otimes W$ and $\mathrm{id}_Z\colon Z \to Z$, we have

$$\mathrm{tr}'_q(\mathrm{id}_Z \otimes f) = \mathrm{id}_Z \otimes \mathrm{tr}'_q(f).$$

(iii) For any morphism $f\colon V \otimes W \to V \otimes W$, we have

$$\mathrm{tr}_q(\mathrm{tr}'_q(f)) = \mathrm{tr}_q(f).$$

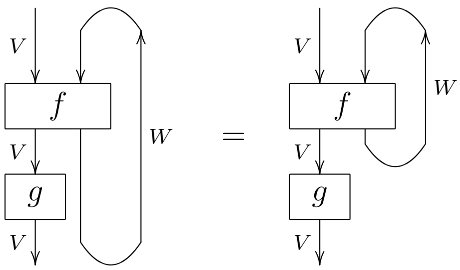


FIGURE 4.16: Property of the partial quantum trace.

Associated to each $U_q(sl_N)$-module V is a natural isomorphism $\theta_V\colon V \to V$ satisfying (Figure 4.17)

$$\theta_V^{\pm 1} = \mathrm{tr}'_q R_{V,V}^{\pm 1}.$$

The naturality (Figure 4.18) means

$$f \circ \theta_V = \theta_W \circ f$$

for $f \in \mathrm{Hom}_{U_q(sl_N)}(V, W)$. In other words, the following diagram commutes

$$\begin{tikzcd} W \arrow[r, "\theta_W"] & W \\ V \arrow[u, "f"] \arrow[r, "\theta_V"'] & V \arrow[u, "f"'] \end{tikzcd}$$

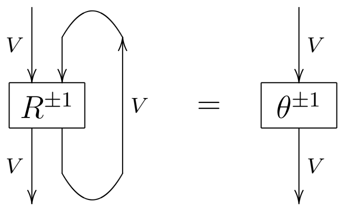


FIGURE 4.17: Properties of θ_V.

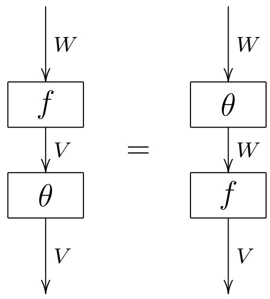


FIGURE 4.18: The naturality of θ_V.

With these objects, we can construct the quantum group invariants of links as follows. Let L be an oriented link with the components $L_1, ..., L_l$ labeled by the $U_q(sl_N)$-modules $V_1, ..., V_l$, respectively. Choose a closed braid representative $\hat{\beta}$ of L with $\beta \in B_n$ being an n-strand braid. Assign to each positive (resp. negative) crossing of β an R-matrix $R_{V,W}$ (resp. $R^{-1}_{W,V}$) where V, W are the $U_q(sl_N)$-modules labeling the two outgoing strands of the crossing (Figure 4.19). Then the braid β

FIGURE 4.19: Crossings and R-matrices.

gives rise to an isomorphism

$$h_{V'_1,...,V'_n}(\beta) \in \mathrm{End}_{U_q(sl_N)}(V'_1 \otimes \cdots \otimes V'_n),$$

where $V'_1, ..., V'_n$ are the $U_q(sl_N)$-modules labeling the strands of β.

If the $U_q(sl_N)$-modules $V_1, ..., V_l$ are irreducible, the isomorphisms $\theta_{V_1}, ..., \theta_{V_l}$ are multiples of the identity maps (by Schur's lemma) and may be regarded as scalars. Let $w(L_i)$ be the *writhe* of L_i in β, i.e. the number of positive crossings minus the number of negative crossings. The *colored HOMFLY polynomial* is defined as

$$I_{L;V_1,...,V_l} = \theta_{V_1}^{-w(L_1)} \cdots \theta_{V_l}^{-w(L_l)} \mathrm{tr}_q[h(\beta)].$$

The fact that these R-matrices satisfy the Yang-Baxter equation guarantees that $h(\beta)$ is a representation of β. Next, we demonstrate the invariance of I under the Markov moves (Figures 4.6 and 4.7). Let's consider the first Markov move: $\beta \leftrightarrow \sigma_i \beta \sigma_i^{-1}$ for $i = 1, ..., n-1$. Clearly the writhes don't change because we add a positive crossing and a negative crossing. Furthermore,

$$\begin{aligned}
\mathrm{tr}_q[h(\sigma_i \beta \sigma_i^{-1})] &= \mathrm{tr}(K^{\otimes n} h(\sigma_i) h(\beta) h(\sigma^{-1})) \\
&= \mathrm{tr}(h(\sigma_i) K^{\otimes n} h(\beta) h(\sigma_i)^{-1}) \\
&= \mathrm{tr}(h(\sigma_i)^{-1} h(\sigma_i) K^{\otimes n} h(\beta)) \\
&= \mathrm{tr}(K^{\otimes n} h(\beta)) = \mathrm{tr}_q[h(\beta)],
\end{aligned}$$

where the second equality follows because $h(\sigma_i)$ is an intertwiner and so it commutes with the action of K, the third equality follows from the properties of trace.

The second Markov move $\beta \leftrightarrow \sigma_n^{\pm 1} \beta$ is trickier. We'll show the case $\beta \leftrightarrow \sigma_n \beta$, the other case is similar. By property (iii) of the partial quantum trace, we have

$$\mathrm{tr}_q[h(\sigma_n \beta)] = \mathrm{tr}_q[\mathrm{tr}'_q(h(\sigma_n \beta))].$$

Now

$$\begin{aligned}
\mathrm{tr}'_q[h(\sigma_n \beta)] &= \mathrm{tr}'_q \left((\mathrm{id}_{V'_1 \otimes \cdots \otimes V'_{n-1}} \otimes R_{V'_n, V'_n}) h(\beta) \right) \\
&= \mathrm{tr}'_q \left(\mathrm{id}_{V'_1 \otimes \cdots \otimes V'_{n-1}} \otimes R_{V'_n, V'_n} \right) h(\beta) \\
&= \left(\mathrm{id}_{V'_1 \otimes \cdots \otimes V'_{n-1}} \otimes \mathrm{tr}'_q(R_{V'_n, V'_n}) \right) h(\beta) \\
&= \left(\mathrm{id}_{V'_1 \otimes \cdots \otimes V'_{n-1}} \otimes \theta_{V'_n} \right) h(\beta),
\end{aligned}$$

where the second equality follows from property (i) of the partial quantum trace, with $f = \mathrm{id}_{V'_1 \otimes \cdots \otimes V'_{n-1}} \otimes R_{V'_n, V'_n}$ and $g = h(\beta)$, the third equality follows from property (ii) of the partial quantum trace and the last equality follows from the property of θ. (The readers are encouraged to draw the diagrams to see what is going on.) Note that $\theta_{V'_n, V'_n}$ is just a scalar and thus can be taken out of the trace, which adds 1 to the writhe of the link component labeled by V_n', leaving I invariant under the second Markov move.

4.5 The Cabling-Projection Rule

For $U_q(sl_N)$-modules V, W, the isomorphism $R_{V,W}$ is usually very complicated, so it is not practical to compute the link invariants from their definition. However, it is known that irreducible $U_q(sl_N)$-modules are always the components of some tensor products of the fundamental $U_q(sl_N)$-module. In this section, we shall follow this observation and develop a *cabling-projection* rule to break down the complexity of general R-matrices.

Let V be the fundamental representation of $U_q(sl_N)$. The *centralizer algebra* of $V^{\otimes n}$ is defined as

$$C_n(V) = \mathrm{End}_{U_q(sl_N)}(V^{\otimes n}) = \{x \in \mathrm{End}(V^{\otimes n}) | xy = yx,\ \forall y \in U_q(sl_N)\}.$$

In other words, $C_n(V)$ consists of all $U_q(sl_N)$-module homomorphisms of $V^{\otimes n}$. A *projection* (or *idempotent*) of $C_n(V)$ is an element $p \in C_n(V)$ satisfying the idempotent equation $p^2 = p$. Let V_λ be the irreducible representation of $U_q(sl_N)$ indexed by $\lambda \vdash n$. There exists a special projection $p_\lambda \in C_n(V)$ (*Young symmetrizer*) such that

$$p_\lambda(V^{\otimes n}) = V_\lambda \subset V^{\otimes n}.$$

Now consider two partitions $\lambda \vdash n$ and $\mu \vdash m$ and let $V_\lambda = p_\lambda(V^{\otimes n})$ and $V_\mu = p_\mu(V^{\otimes m})$ be the irreducible $U_q(sl_N)$-modules indexed by λ and μ, respectively. By naturality of R-matrices (Figure 4.9), we have

$$R_{V_\lambda,V_\mu} \circ (p_\lambda \otimes p_\mu) = (p_\mu \otimes p_\lambda) \circ R_{V^{\otimes n},V^{\otimes m}}.$$

By repeatedly applying the triangle equalities (Figures 4.10 and 4.11), we have

$$R_{V^{\otimes n},V^{\otimes m}} = h_V(\beta_{n,m}),$$

where $\beta_{n,m}$ is the braid obtained by cabling the strand labeled by $V^{\otimes n}$ to n parallel ones and the strand labeled by $V^{\otimes m}$ to m parallel ones (Figure 4.20). Thus we obtain the following cabling-projection rule

$$R_{V_\lambda,V_\mu} \circ (p_\lambda \otimes p_\mu) = (p_\mu \otimes p_\lambda) \circ h_V(\beta_{n,m}).$$

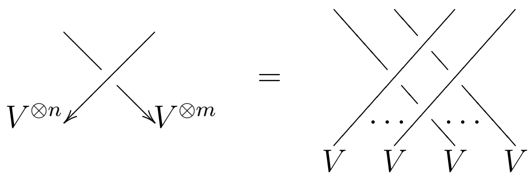


FIGURE 4.20: The cabling rule.

Recall that if $f\colon B \to A \subset B$ is a linear transformation of the vector space B, then

$$\mathrm{tr}(f) = \mathrm{tr}(f\,|_A),$$

where $f\,|_A$ denotes the restriction of f to the subspace A. Let $\beta \in B_n$ be a closed braid representative of some link L with the strands labeled by $V_i = p_{\lambda^i}(V^{\otimes|\lambda^i|})$, $i = 1, ..., n$. We have the following map

$$\phi\colon = K^{\otimes n} \circ h_{V_1,\ldots,V_n}(\beta) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n})$$

is a linear transformation from $V^{\otimes|\lambda^1|} \otimes \cdots \otimes V^{\otimes|\lambda^n|}$ to $V_1 \otimes \cdots \otimes V_n$, considered as a subspace of $V^{\otimes|\lambda^1|} \otimes \cdots \otimes V^{\otimes|\lambda^n|}$. Thus,

$$\mathrm{tr}_q(h_{V_1,\ldots,V_n}(\beta) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n})) = \mathrm{tr}_q(h_{V_1,\ldots,V_n}(\beta)).$$

(The fact that the restriction of ϕ to $V_1 \otimes \cdots \otimes V_n$ is $K^{\otimes n} \circ h_{V_1,\ldots,V_n}(\beta)$ follows from the idempotent property of p_{λ^i}.) By repeating applications of the cabling-projection rule to ϕ we obtain

$$\phi = K^{\otimes n} \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n}) \circ h_V(\beta_{m_1,\ldots,m_n}),$$

where we let $m_i = |\lambda^i|$, $i = 1, ..., n$. Now since p_{λ^i} is a module homomorphism, it commutes with the action of K. Thus

$$\phi = (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n}) \circ K^{\otimes n} \circ h_V(\beta_{m_1,\ldots,m_n}).$$

And so

$$\begin{aligned}
\mathrm{tr}_q(h_{V_1,\ldots,V_n}(\beta) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n})) &= \mathrm{tr}(\phi) \\
&= \mathrm{tr}((p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n}) \circ K^{\otimes n} \circ h_V(\beta_{m_1,\ldots,m_n})) \\
&= \mathrm{tr}(K^{\otimes n} \circ h_V(\beta_{m_1,\ldots,m_n}) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n})) \\
&= \mathrm{tr}_q(h_V(\beta_{m_1,\ldots,m_n}) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^n})).
\end{aligned}$$

Hence we have transformed the problem of calculating the colored HOMFLY polynomial over arbitrary irreducible representations of $U_q(sl_N)$ to the problem of calculating the colored HOMFLY polynomial over the fundamental representation of $U_q(sl_N)$. In summary, given some closed braid representative β with m strands, where the i^{th} strand is labeled with V_{λ^i} for some partition $\lambda^i \vdash n_i$ (note that the strands which belong to the same link component are labeled by the same representation), to calculate the quantum trace of $h(\beta)$ over $V_{\lambda^1} \otimes \cdots \otimes V_{\lambda^m}$, we proceed as follows.

- Putting the projection p_{λ^i} underneath the i^{th} strand of β.
- Cabling the i^{th} strand of β to n_i parallel ones to obtain the cabling braid $\beta_{n_1,\ldots,n_m}$.
- Calculating the quantum trace of $h_V(\beta_{n_1,\ldots,n_m}) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^m})$ over $V^{\otimes n}$, where $n = n_1 + \cdots + n_m$.

4.6 The Centralizer Algebras

Given a closed braid representative β of some link L, we can replace β by its cabling braid together with the projections via the cabling-projection rule, giving rise to an intertwiner of $V^{\otimes n}$, which is an element of $C_n(V)$. It is a standard result that the $U_q(sl_N)$-module $V^{\otimes n}$ has the following decomposition

$$V^{\otimes n} = \bigoplus_{j=1}^{k} \bigoplus_{i=1}^{d_{\lambda_j}} V^i_{\lambda^j},$$

where λ^j runs through the set of partitions of n and $V^i_{\lambda^j}$ denotes the irreducible representation of $U_q(sl_N)$ indexed by λ^j. We identify $V^i_{\lambda^j}$ with a subspace of $V^{\otimes n}$ by the projection map

$$p^i_{\lambda^j} : V^{\otimes n} \to V^i_{\lambda^j} \subset V^{\otimes n}.$$

Choose the same basis $\{v^{\lambda^j}_{i,1}, ..., v^{\lambda^j}_{i,m_{\lambda^j}}\}$ for each $V^i_{\lambda^j}$. A basis for $V^{\otimes n}$ is obtained by concatenating the bases of $V^i_{\lambda^j}$ in the order of the appearing factors. With respect to that basis, an element $x \in C_n(V)$ has the matrix representation of the form given in Figure 4.21, where $A^{i,j}_{\lambda^r,\lambda^s}$ is the matrix corresponding to $V^i_{\lambda^r}$ and $V^j_{\lambda^s}$. By the idempotent property, each $p^i_{\lambda^j}$ acts as an identity on $V^i_{\lambda^j}$. Thus the matrix representation of $p^i_{\lambda^j}$ is the identity at the block labeled $A^{i,i}_{\lambda^j,\lambda^j}$ and zero everywhere else. Note that an arbitrary block $A^{i,j}_{\lambda^r,\lambda^s}$ of x can be obtained by

$$p^i_{\lambda^r} \circ x \circ p^j_{\lambda^s} : V^j_{\lambda^s} \to V^i_{\lambda^r}.$$

$$\begin{pmatrix}
A^{1,1}_{\lambda^1,\lambda^1} & \cdots & A^{1,d_{\lambda^1}}_{\lambda^1,\lambda^1} & & A^{1,1}_{\lambda^1,\lambda^k} & \cdots & A^{1,d_{\lambda^k}}_{\lambda^1,\lambda^k} \\
\vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\
A^{d_{\lambda^1},1}_{\lambda^1,\lambda^1} & \cdots & A^{d_{\lambda^1},d_{\lambda^1}}_{\lambda^1,\lambda^1} & & A^{d_{\lambda^1},1}_{\lambda^1,\lambda^k} & \cdots & A^{d_{\lambda^1},d_{\lambda^k}}_{\lambda^1,\lambda^k} \\
 & \vdots & & \ddots & & \vdots & \\
A^{1,1}_{\lambda^k,\lambda^1} & \cdots & A^{1,d_{\lambda^1}}_{\lambda^k,\lambda^1} & & A^{1,1}_{\lambda^k,\lambda^k} & \cdots & A^{1,d_{\lambda^k}}_{\lambda^k,\lambda^k} \\
\vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\
A^{d_{\lambda^k},1}_{\lambda^k,\lambda^1} & \cdots & A^{d_{\lambda^k},d_{\lambda^1}}_{\lambda^k,\lambda^1} & & A^{d_{\lambda^k},1}_{\lambda^k,\lambda^k} & \cdots & A^{d_{\lambda^k},d_{\lambda^k}}_{\lambda^k,\lambda^k}
\end{pmatrix}$$

FIGURE 4.21: The matrix form of x.

By Schur's lemma, $A^{i,j}_{\lambda^r,\lambda^s} = 0$ unless $r = s$. For the case $r = s$, we have

$$A^{i,j}_{\lambda^r,\lambda^r} = c^{i,j}_{\lambda^r}\mathrm{id}.$$

and can be identified with a scalar. Consequently,

$$C_n(V) = \bigoplus_{j=1}^{k} \mathrm{Mat}_{d_{\lambda^j}}(\mathbb{C}),$$

where $\mathrm{Mat}_{d_{\lambda^j}}(\mathbb{C})$ denotes the full matrix algebra of $d_{\lambda^j} \times d_{\lambda^j}$ matrices over $\mathbb{C}$. By general representation theory, see for example [5], the irreducible represetations of $C_n(V)$ are naturally indexed by partitions of n and the irreducible character corresponding to λ^j is the trace of the block labeled λ^j.

The action of the special element K in $U_q(sl_N)$ gives rise to a linear operator on $V^{\otimes n}$ (it may not belong to the centralizer algebra). The matrix representation of K is block-diagonal and is given in Figure 4.22, where $K^{i,i}_{\lambda^j,\lambda^j}$ is the matrix corresponding to the action of K on $V^i_{\lambda^j}$. (Since K is an action, $K^{i,j}_{\lambda^r,\lambda^s} = 0$ unless $r = s$ and $i = j$.) By our choice of basis, $K_{\lambda^j} := K^{1,1}_{\lambda^j,\lambda^j} = ... = K^{d_{\lambda^j},d_{\lambda^j}}_{\lambda^j,\lambda^j}$ for

$$\begin{pmatrix}
K^{1,1}_{\lambda^1,\lambda^1} & \cdots & 0 & & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\
0 & \cdots & K^{d_{\lambda^1},d_{\lambda^1}}_{\lambda^1,\lambda^1} & & 0 & \cdots & 0 \\
 & \vdots & & \ddots & & \vdots & \\
0 & \cdots & 0 & & K^{1,1}_{\lambda^k,\lambda^k} & \cdots & 0 \\
\vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & & 0 & \cdots & K^{d_{\lambda^k},d_{\lambda^k}}_{\lambda^k,\lambda^k}
\end{pmatrix}$$

FIGURE 4.22: The matrix form of K.

$j = 1, .., k$. Summarizing our observations so far, we have

$$\mathrm{tr}_q(x) = \mathrm{tr}(K^{\otimes n} \circ x) = \sum_{j=1}^{k} \sum_{i=1}^{d_{\lambda^j}} c_{\lambda^j}^{i,i} \mathrm{tr}(K_{\lambda^j}).$$

Note that the term

$$\mathrm{tr}(K_{\lambda^j}) = \mathrm{tr}_q(V_{\lambda^j}),$$

is called the *quantum dimension* of V_{λ^j} and denoted by $\dim_q(V_{\lambda^j})$.

4.7 Connection to the Hecke Algebra

For the fundamental representation V of $U_q(sl_N)$, the action of the R-matrix has a special form. With suitable basis $\{v_1, ..., v_N\}$ of V, we have

$$q^{1/N} R_{V,V}(v_i \otimes v_j) = \begin{cases} q v_i \otimes v_j, & i = j, \\ v_j \otimes v_i, & i < j, \\ v_j \otimes v_i + (q - q^{-1}) v_i \otimes v_j, & i > j. \end{cases}$$

It is not difficult to check that $q^{1/N} R_{V,V}$ satisfies the quadratic relation in the Hecke algebra $H_n(q)$

$$g_i^2 + (q^{-1} - q) g_i - 1 = 0, \quad i = 1, ..., n-1.$$

Thus we obtain a representation of the Hecke algebra. Let $\tilde{h}_V \colon H_n(q) \to C_n(V)$ be the homomorphism given by

$$\tilde{h}_V(g_i) = \begin{cases} q^{1/N} R_{V,V} \otimes \mathrm{id}_{V^{\otimes(n-2)}}, & i = 1, \\ \mathrm{id}_{V^{\otimes(i-1)}} \otimes q^{1/N} R_{V,V} \otimes \mathrm{id}_{V^{\otimes(n-i-1)}}, & 1 < i < n-1, \\ \mathrm{id}_{V^{\otimes(n-2)}} \otimes q^{1/N} R_{V,V}, & i = n-1. \end{cases}$$

Let β be a closed braid representative with m strands, where the i^{th} strand is labeled with V_{λ^i} for some partition $\lambda^i \vdash n_i$ and let $n = n_1 + \cdots + n_m$, the following map

$$x \colon = \tilde{h}_V(\beta_{n_1,...,n_m}) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^m}) \colon V^{\otimes n} \to V^{\otimes n}$$

is clearly an element of $C_n(V)$. By the result in the previous section, we have

$$\mathrm{tr}_q(x) = \sum_{j=1}^{k} \sum_{i=1}^{d_{\lambda^j}} c_{\lambda^j}^{i,i} \dim_q(V_{\lambda^j}),$$

where λ^j runs through the set of partitions of n. It turns out that the trace of x over the block labeled by λ^j is precisely the irreducible character ζ^{λ^j} of the Hecke algebra [7] evaluated at

$$b\colon = \beta_{n_1,\dots,n_m} \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^m}).$$

(We shall explain how to view the projection maps as elements of the Hecke algebra bellow.) On the other hand,

$$\begin{aligned}
\mathrm{tr}_q(x) &= q^{w(\beta_{n_1,\dots,n_m})/N} \mathrm{tr}_q(h_V(\beta_{n_1,\dots,n_m}) \circ (p_{\lambda^1} \otimes \cdots \otimes p_{\lambda^m})) \\
&= q^{w(\beta_{n_1,\dots,n_m})/N} \mathrm{tr}_q[h_{V_{\lambda^1},\dots,V_{\lambda^m}}(\beta)],
\end{aligned}$$

where $w(\beta_{n_1,\dots,n_m})$ denotes the writhe of $\beta_{n_1,\dots,n_m}$ (the number of positive crossings minus the number of negative crossings). Thus we obtain

$$\mathrm{tr}_q[h_{V_{\lambda^1},\dots,V_{\lambda^m}}(\beta)] = q^{-w(\beta_{n_1,\dots,n_m})/N} \sum_{\lambda \vdash n} \zeta^\lambda(b) \dim_q(V_\lambda).$$

It turns out that the projection maps can be realized as images of certain elements in the Hecke algebra [2]. Speciafically, for any partition $\lambda \vdash n$, there exists $y_\lambda \in H_n(q)$ (also known as *Young symmetrizer*) such that

$$p_\lambda = \tilde{h}_V(y_\lambda).$$

In order to construct y_λ, we first need the following elements in the Hecke algebra:

$$f_n = \sum_{\sigma \in S_n} q^{-\frac{n(n-1)}{2}+l(\sigma)} T_\sigma,$$

$$g_n = \sum_{\sigma \in S_n} (-1)^{-l(\sigma)} q^{\frac{n(n-1)}{2}-l(\sigma)} T_\sigma,$$

where $l(\sigma)$ denotes the length of the permutation σ, i.e., the number of factors in the reduced decomposition of σ (it is precisely the number of crossings of T_σ). We call f_n the *symmetrizer* and g_n the *anti-symmetrizer*. Given a Young diagram $\lambda = (\lambda_1, ..., \lambda_k)$, the *transpose* of λ, λ^t, is defined as

$$\lambda^t = (\lambda^t_1, ..., \lambda^t_l),$$

where λ^t_j is the number of boxes in the j^{th} column of λ. Let $\lambda(i,j)$ denote the box at position (i,j) (row i, column j) of the Young diagram, its *hook length* is defined to be

$$hl(\lambda(i,j)) = \lambda_i + \lambda_j^t - i - j + 1.$$

For an integer n, the quantum integer $[n]_q$ is defined as

$$[n]_q :\ = \frac{q^n - q^{-n}}{q - q^{-1}}.$$

The *quantum hook length* of λ is

$$[hl(\lambda)]_q = \prod [hl(\lambda(i,j))]_q,$$

where the product is over all boxes of λ. Next, we put

$$s = f_{\lambda_k} \otimes \cdots \otimes f_{\lambda_1},$$

$$a = g_{\lambda_1^t} \otimes \cdots \otimes g_{\lambda_l^t},$$

and connect them using some special braid b (Figure 4.23). Finally,

$$y_\lambda = \frac{1}{[hl(\lambda)]} b^{-1} sba.$$

We still need a few more ingredients in the calculation of the colored HOMFLY polynomial. For each partition $\lambda \vdash n$ with $l(\lambda) \leq N$ (the number of parts of λ), the twist θ_{V_λ} is given by

$$\theta_{V_\lambda} = q^{\kappa_\lambda + nN - n^2 N} \mathrm{id}_{V_\lambda},$$

where

$$\kappa_\lambda = \sum_{i=1}^{l(\lambda)} \sum_{j=1}^{\lambda_i} 2(j-i).$$

And the quantum dimesion of V_λ is given by

$$\dim_q(V_\lambda) = s_\lambda\Big(q^{N-1}, q^{N-3}, ..., q^{-(N-1)}\Big),$$

the Schur polynomial corresponding to λ in the variables $q^{N-1}, ..., q^{-(N-1)}$.

Now we can summarize the calculation of the colored HOMFLY polynomial in the next theorem.

Theorem 4.1 ([9]). *Let L be an oriented link with l components $L_1, ..., L_l$ labeled by irreducible $U_q(sl_N)$-modules $V_{\lambda^1}, ..., V_{\lambda^l}$ for partitions $\lambda^i \vdash n_i$, respectively. Suppose $\beta \in B_m$ is a closed braid representative of L and the m strands of β are living on*

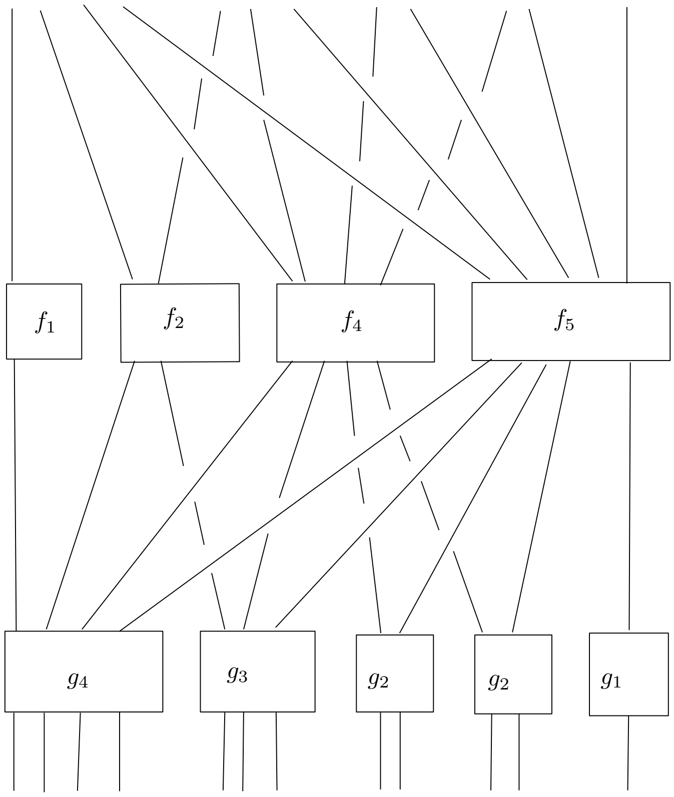


FIGURE 4.23: The Young symmetrizer corresponds to the partition (5,4,2,1).

$L_{i_1}, ..., L_{i_m}$, *respectively. Then*

$$I_{L;V_{\lambda^1},\dots,V_{\lambda^l}} = q^{-\sum_{i=1}^{l}(\kappa_{\lambda^i}+n_i-n_i^2/N)w(L_i)-w(\beta_{n_{i_1},\dots,n_{i_m}})/N} \cdot \sum_{\lambda\vdash n} \zeta^{\lambda}(b)\cdot s_{\lambda}\left(q^{N-1}, q^{N-3}, \dots, q^{-(N-1)}\right),$$

where $n = n_{i_1} + \cdots + n_{i_m}$, $\beta_{n_{i_1},\dots,n_{i_m}}$ *is the braid obtained by cabling the* j^{th} *strand of* β *to* n_{i_j} *parallel ones and* $b = \beta_{n_{i_1},\dots,n_{i_m}} \circ (p_{\lambda^{i_1}} \otimes \cdots \otimes p_{\lambda^{i_m}})$.

Note that in the formula of $I_{L;V_{\lambda^1},\dots,V_{\lambda^l}}$, there is an explicit factor $q^{1/N}$ to the power

$$\sum_{i=1}^{l} n_i^2 w(L_i) - w(\beta_{n_{i_1},\dots,n_{i_m}}).$$

We drop this insignificant factor and regard the remaining part as a rational function of q and q^N (here we think of N as a variable). Let $W_{L;\lambda^1,\dots,\lambda^l}(t,\nu)$ be the remaining part of $I_{L;V_{\lambda^1},\dots,V_{\lambda^l}}$ with q replaced by $t^{-1/2}$ and q^N replaced by $\nu^{-1/2}$. Thus

$$W_{L;\lambda^1,\dots,\lambda^l}(t,\nu) = t^{\sum_{i=1}^{l}\kappa_{\lambda^i}w(L_i)/2}\nu^{\sum_{i=1}^{l}n_i w(L_i)/2} \sum_{\lambda\vdash n} \zeta^{\lambda}(x)\big|_{q=t^{-1/2}} s_{\lambda}^{*}(t,\nu),$$

where

$$s^*_\lambda(t,\nu) = s_\lambda\Big(q^{N-1},...,q^{-(N-1)}\Big)\Big|_{q=t^{-1/2},q^N=\nu^{-1/2}}.$$

From now on, we shall refer to $W_{L;\lambda^1,...,\lambda^l}(t,\nu)$ as the (normalized) colored HOMFLY polynomial. By some results on symmetric functions [9], the Schur polynomials can be expressed as

$$s^*_\lambda(t,\nu) = \sum_{\mu\vdash n}\frac{|C_\mu|}{n!}\chi^\lambda(C_\mu)\prod_{i=1}^{l(\mu)}\frac{\nu^{\mu_i/2}-\nu^{-\mu_i/2}}{t^{\mu_i/2}-t^{-\mu_i/2}},$$

where C_μ denote the conjugacy class of the symmetric group S_n indexed by $\mu\vdash n$ and χ^λ denote the character of the symmetric group S_n corresponding to λ. To obtain χ^λ, we just need to set $q=1$ in our algorithm to compute the characters of the Hecke algebra.

Chapter 5

The LMOV Conjecture

5.1 Computing the Colored HOMFLY Polynomial

We've developed all the machinery. In this section, we'll calculate the colored HOMFLY polynomial of the figure 8 knot (Figure 5.1) labeled by $\lambda = (1)$ (the Young diagram consisting of one box, corresponding to the fundamental representation), $\lambda = (2)$ (the Young diagram consisting of two horizontal boxes) and $\lambda = (1, 1)$ (the Young diagram consisting of two vertical boxes), respectively. The

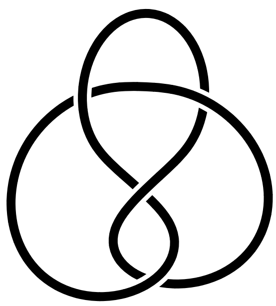
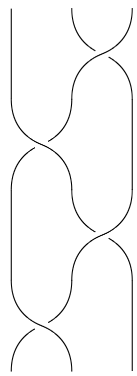


FIGURE 5.1: The figure 8 knot.

figure 8 knot has the following closed braid representative.



If we label the figure 8 knot with $\lambda = (1)$, then we obtain the following colored HOMFLY polynomial:

$$W_{(1)}(t,\nu) = \frac{(-1+\nu)(-\nu - t^2\nu + t(1+\nu+\nu^2))}{(-1+t)\sqrt{t}\nu^{3/2}}.$$

Note that we have

$$\frac{t^{1/2} - t^{-1/2}}{\nu^{1/2} - \nu^{-1/2}} W_{(1)}(t,\nu) = 1 - \frac{1}{t} - t + \frac{1}{v} + v,$$

which is the HOMFLY polynomial, as expected. If we label the figure 8 knot with $\lambda = (2)$, then we obtain the following colored HOMFLY polynomial:

$$\begin{aligned} W_{(2)}(t,\nu) = & \frac{1}{(-1+t)^2 t^2 (1+t)\nu^3} \big[-(-1+\nu)^2\nu - t^2(-1+\nu)\nu^2 \\ & + t^5(-1+\nu)\nu^3 - t^7(-1+\nu)^2\nu^3 \\ & + t^3\nu^2(2 - 2\nu + \nu^2 - \nu^3) \\ & + t^4\nu(-1+\nu-2\nu^2+2\nu^3) \\ & + t^6\nu^2(1-\nu+\nu^2-2\nu^3+\nu^4) \\ & + t(1-2\nu+\nu^2-\nu^3+\nu^4) \big]. \end{aligned}$$

If we label the figure 8 knot with $\lambda = (1,1)$, then we obtain the following colored HOMFLY polynomial

$$
\begin{aligned}
W_{(1,1)}(t,\nu) = \frac{1}{(-1+t)^2t^2(1+t)\nu^3}\big[&-(-1+\nu)^2\nu^3 + t^2(-1+\nu)\nu^3 \\
&- t^5(-1+\nu)\nu^2 - t^7(-1+\nu)^2\nu \\
&+ t^3\nu(-1+\nu-2\nu^2+2\nu^3) \\
&+ t^4\nu^2(2-2\nu+\nu^2-\nu^3) \\
&+ t^6(1-2\nu+\nu^2-\nu^3+\nu^4) \\
&+ t\nu^2(1-\nu+\nu^2-2\nu^3+\nu^4)\big].
\end{aligned}
$$

Note that the coefficients of t^n in $W_{(1,1)}$ is exactly the same as in $W_{(2)}$ but in the reverse order. In other words, we have the following permutation of coefficients of t^n for $n = 0, 1, ..., 7$:

$$
\begin{pmatrix}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
7 & 6 & 5 & 4 & 3 & 2 & 1 & 0
\end{pmatrix}.
$$

5.2 The LMOV Conjecture

Given a knot L, consider the following series:

$$
\begin{aligned}
Z(t,\nu) &= 1 + W_{(1)}(t,\nu)s_{(1)} + W_{(2)}s_{(2)} + W_{(1,1)}s_{(1,1)} + \cdots \\
&= \sum_{\lambda} W_\lambda s_\lambda,
\end{aligned}
$$

where the sum runs over all partitions, including the empty one (corresponds to 1) and s_λ is the Schur polynomial corresponding to λ. Recall that the Möbius function of a positive integer n is given by

$$
\mu(n) = \begin{cases} (-1)^r & \text{if } n = p_1 \cdots p_r \text{ and the } p_i \text{ are distinct primes,} \\ 0 & \text{otherwise.} \end{cases}
$$

Using the formula

$$
\log Z = -(1-Z) - \frac{(1-Z)^2}{2} - \frac{(1-Z)^3}{3} - \cdots = -\sum_{n=1}^{\infty} \frac{(1-Z)^n}{n},
$$

and the multiplication rule for Schur polynomials, we can express the following series as

$$\sum_{n=1}^{\infty} \frac{\mu(n)}{n} \log Z(t^n, \nu^n) = \sum_{\lambda} f_\lambda(t, \nu) s_\lambda.$$

The functions $f_\lambda(t, \nu)$ are refered to as the *reformulated colored HOMFLY polynomials*. For instance, a first few f_λ are

$$\begin{aligned}
\sum_{n=1}^{\infty} \frac{\mu(n)}{n} \log Z(t^n, \nu^n) &= W_{(1)}(t,\nu)s_{(1)} + \left(W_{(2)}(t,\nu) - \frac{W_{(1)}(t,\nu)^2}{2} - \frac{W_{(1)}(t^2,\nu^2)}{2}\right) s_{(2)} \\
&+ \left(W_{(1,1)}(t,\nu) - \frac{W_{(1)}(t,\nu)^2}{2} + \frac{W_{(1)}(t^2,\nu^2)}{2}\right) s_{(1,1)} \\
&+ \left(W_{(3)}(t,\nu) - W_{(1)}(t,\nu)W_{(2)}(t,\nu) + \frac{W_{(1)}(t,\nu)^3}{3} - \frac{W_{(1)}(t^3,\nu^3)}{3}\right) s_{(3)} \\
&+ \left(W_{(2,1)}(t,\nu) - W_{(1)}(t,\nu)(W_{(2)}(t,\nu) + W_{(1,1)}(t,\nu)) + \frac{2}{3}W_{(1)}(t,\nu)^3 + \frac{W_{(1)}(t^3,\nu^3)}{3}\right) s_{(2,1)} \\
&+ \left(W_{(1,1,1)}(t,\nu) - W_{(1)}(t,\nu)W_{(1,1)}(t,\nu) + \frac{W_{(1)}(t,\nu)^3}{3} - \frac{W_{(1)}(t^3,\nu^3)}{3}\right) s_{(1,1,1)} + \cdots
\end{aligned}$$

For $\lambda, \mu \vdash n$, let

$$M_{\lambda\mu}(t) = \sum_{\tau \vdash n} \frac{|C_\tau|}{n!} \chi^\lambda(C_\tau)\chi^\mu(C_\tau) \frac{\prod_{j=1}^{l(\tau)}(t^{-\tau_j/2} - t^{\tau_j/2})}{t^{-1/2} - t^{1/2}}.$$

The LMOV conjecture says that these f_λ have the following highly nontrivial structure:

$$f_\lambda(t,\nu) = \sum_{\mu \vdash |\lambda|} \hat{f}_\mu(t,\nu) M_{\lambda\mu}(t),$$

where the $\hat{f}_\mu$ have the following *genus expansion*

$$\hat{f}_\mu(t,\nu) = \sum_{g \geq 0} \sum_{Q} N_{\mu,g,Q}(t^{1/2} - t^{-1/2})^{2g-1}\nu^Q,$$

where Q are either all integers or all semi-integers and $N_{\mu,g,Q}$ are all integers. Moreover, the integers $N_{\mu,g,Q}$ can be interpreted in terms of the enumerative geometry of Riemann surfaces with boundaries in a certain Calabi-Yau threefold [8].

We will test the LMOV conjecture for the figure 8 knot labeled with $\lambda = (1)$, $\lambda = (2)$, $\lambda = (1,1)$, respectively. Recall that we have

$$f_{(1)}(t,\nu) = W_{(1)}(t,\nu),$$

$$f_{(2)}(t,\nu) = W_{(2)}(t,\nu) - \frac{W_{(1)}(t,\nu)^2}{2} - \frac{W_{(1)}(t^2,\nu^2)}{2},$$
$$f_{(1,1)}(t,\nu) = W_{(1,1)}(t,\nu) - \frac{W_{(1)}(t,\nu)^2}{2} + \frac{W_{(1)}(t^2,\nu^2)}{2}.$$

Note that for the Young diagram with one box, we have $M_{(1),(1)} = 1$. Thus

$$f_{(1)}(t,\nu) = \hat{f}_{(1)}(t,\nu) = W_{(1)}(t,\nu).$$

Put $T = t^{1/2} - t^{-1/2}$, we have

$$\begin{aligned} W_{(1)}(t,\nu) &= \left(\frac{\nu^{1/2}-\nu^{-1/2}}{t^{1/2}-t^{-1/2}}\right)(1 - t^{-1} - t + \nu^{-1} + \nu) \\ &= \left(\frac{\nu^{1/2}-\nu^{-1/2}}{T}\right)(-1 - T^2 + \nu^{-1} + \nu) \\ &= (-\nu^{-3/2} + 2\nu^{-1/2} - 2\nu^{1/2} + \nu^{3/2})T^{-1} + (\nu^{-1/2} - \nu^{1/2})T, \end{aligned}$$

which has the required form. For the Young diagrams with two boxes, we have

$$\begin{aligned} M_{(2),(2)}(t) &= \frac{|C_{(2)}|}{2}\chi^{(2)}(C_{(2)})\chi^{(2)}(C_{(2)})\left(\frac{t^{-1}-t}{t^{-1/2}-t^{1/2}}\right) \\ &\quad + \frac{|C_{1,1}|}{2}\chi^{(2)}(C_{(1,1)})\chi^{(2)}(C_{(1,1)})\left(\frac{(t^{-1/2}-t^{1/2})^2}{t^{-1/2}-t^{1/2}}\right) \\ &= \frac{1}{2}(t^{-1/2}+t^{1/2}) + \frac{1}{2}(t^{-1/2}-t^{1/2}) = t^{-1/2}. \end{aligned}$$

Similarly,

$$M_{(2),(1,1)}(t) = -t^{1/2}, \quad M_{(1,1),(2)}(t) = -t^{1/2}, \quad M_{(1,1),(1,1)}(t) = t^{-1/2}.$$

Thus we obtain

$$\begin{cases} f_{(2)}(t,\nu) = t^{-1/2}\hat{f}_{(2)}(t,\nu) - t^{1/2}\hat{f}_{(1,1)}(t,\nu), \\ f_{(1,1)}(t,\nu) = -t^{1/2}\hat{f}_{(2)}(t,\nu) + t^{-1/2}\hat{f}_{(1,1)}(t,\nu). \end{cases}$$

Solving the equations we get

$$\begin{aligned} \hat{f}_{(2)}(t,\nu) &= \frac{1}{(t^{-1}-t)t^{1/2}}\left(W_{(2)}(t,\nu) - \frac{W_{(1)}^2(t,\nu)}{2} - \frac{W_{(1)}(t^2,\nu^2)}{2}\right) \\ &\quad + \frac{t^{1/2}}{t^{-1}-t}\left(W_{(1,1)}(t,\nu) - \frac{W_{(1)}^2(t,\nu)}{2} + \frac{W_{(1)}(t^2,\nu^2)}{2}\right), \end{aligned}$$

and

$$\hat{f}_{(1,1)}(t,\nu) = \frac{1}{(t^{-1}-t)t^{1/2}}\left(W_{(1,1)}(t,\nu) - \frac{W^2_{(1)}(t,\nu)}{2} + \frac{W_{(1)}(t^2,\nu^2)}{2}\right)$$
$$+ \frac{t^{1/2}}{t^{-1}-t}\left(W_{(2)}(t,\nu) - \frac{W^2_{(1)}(t,\nu)}{2} - \frac{W_{(1)}(t^2,\nu^2)}{2}\right).$$

By some algebraic manipulations, we obtain

$$\begin{aligned}\hat{f}_{(2)}(t,\nu) &= (-2\nu^{-3} + 7\nu^{-2} - 9\nu^{-1} + 6 - 4\nu + 3\nu^2 - \nu^3)T^{-1} \\ &+ (-\nu^{-3} + 4\nu^{-2} - 5\nu^{-1} + 3 - 2\nu + \nu^2)T \\ &+ (\nu^{-2} - 2\nu^{-1} + 1)T^3,\end{aligned}$$

and

$$\begin{aligned}\hat{f}_{(1,1)}(t,\nu) &= (-\nu^{-3} + 3\nu^{-2} - 4\nu^{-1} + 6 - 9\nu + 7\nu^2 - 2\nu^3)T^{-1} \\ &+ (\nu^{-2} - 2\nu^{-1} + 3 - 5\nu + 4\nu^2 - \nu^3)T \\ &+ (1 - 2\nu + \nu^2)T^3,\end{aligned}$$

as expected.

5.3 Conclusion

In 2007 there was a proposed proof for the LMOV conjecture for the case of sl_N [10]. Further studies should be done on other simple Lie algebras, say the Lie algebra so_N. Also attempts should be made to compute these integers $N_{g,Q}$ appear in the reformulated colored HOMFLY polynomial topologically. The LMOV conjecture establishes a connection between two important physical theories: Chern-Simons gauge theory and topological string theory. It also helps us understand more about the basic structure of link invariants and three-manifold invariants.

Bibliography

[1] Colin C. Adams. *The Knot Book*. Freeman, 1994.

[2] Christian Blanchet. Hecke Algebras, Modular Categories and 3-Manifolds Quantum Invariants. *Topology*, 39:193–223, 2000.

[3] D.S. Dummit and R.M. Foote. *Abstract Algebra*. Wiley, third edition, 2004.

[4] Pavel Etingof, Oleg Golberg, Sebastian Hensel, Tiankai Liu, Alex Schwendner, Elena Udovina, and Dmitry Vaintrob. Introduction to Representation Theory, 2009. URL http://arxiv.org/abs/0901.0827.

[5] William Fulton. *Representation Theory: a First Course*. Springer-Verlag, 1991.

[6] William Fulton. *Young Tableaux*. Cambridge University Press, 1997.

[7] M. Jimbo. A q-Analogue of $U(gl(N+1))$, Hecke Algebra, and the Yang-Baxter Equation. *Lett. Math. Phys.*, 11:247–252, 1986.

[8] José M.F. Labastida and Marcos Mariño. A New Point of View in the Theory of Knot and Link Invariants. 2001. URL http://arxiv.org/abs/math/0104180v4.

[9] Xiao-Song Lin and Hao Zheng. On the Hecke Algebras and the Colored HOMFLY Polynomial. 2006. URL http://arxiv.org/abs/math/0601267v1.

[10] Kefeng Liu and Pan Peng. Proof of the Labastida-Mariño-Ooguri-Vafa Conjecture. 2007. URL http://arxiv.org/abs/0704.1526v3.

[11] Arun Ram. A Frobenius Formula for the Characters of the Hecke Algebras. *Invent. math.*, 106:461–488, 1991.

[12] Arun Ram and J.B. Remmel. Applications of the Frobenius Formulas for the Characters of the Symmetric Group and the Hecke Algebras of Type A. *Journal of Algebraic Combinatorics*, 6:59–87, 1997.

[13] J.P. Serre. *Linear Representations of Finite Groups*. Springer, 1977.

[14] V.G. Turaev. *Quantum Invariants of Knots and 3-Manifolds*. W. de Gruyter, 1994.

[15] Chari Vyjayanthi and Andrew Pressley. *A Guide to Quantum Groups*. Cambridge University Press, 2005.

[16] Wikipedia. Perko pair — wikipedia, the free encyclopedia, 2008. URL http://en.wikipedia.org/w/index.php?title=Perko_pair&oldid=201606471. [Online; accessed 21-October-2009].