BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI, INDIA
FUDAN UNIVERSITY, SHANGHAI, CHINA

Abstract

Prof. Ling-Yan Hung, *
Dr. Prasanna Kumar N. **
* Department of Physics, Fudan University,
** Department of Mathematics, BITS, Pilani.

Master of Science, Mathematics

Boundary conditions of Levin-Wen type models

by Amit Jamadagni

Topological Phases of Matter at absolute zero cannot be classified by Group theoretical approach. The thesis aims to present the models which classify these phases of matter and further present various properties of these models. The construction of the models assumes a lattice structure, with edges and nodes being represented in different ways for different models though they form a subclass of the String-Net model by Levin-Wen, according to which the edges come from a Unitary Tensor Category, the excitations form a Modular Tensor Category. The lattice with boundary is also considered with the boundaries given by the modules over algebras of the Modular Tensor Category. The introduction is through the Kitaev Quantum Double Models, the celebrated Toric Code is one such model, which later are expressed in terms of Categorial parlence. Excitations, along with that condense on the given boundary, are discussed for the Quantum Double of S_3. Ribbon operators which carry excitations at the end are evaluated in the absence and presence of boundaries giving rise to the ribbon operators which represent anyon condensation in the latter, this leading to the construction of ground states. The identification of boundaries in terms of Category theory is also presented in terms of theorems. The experience from Quantum Doubles allows one to construct (identifying the boundaries from the theorem) and verify the ground states for Ising⊠Ising with boundary as Ising, in the absense of ribbon operator. Finally, the construction of ribbon operators analogue, the string operator is presented with an aim to identify the string operators connecting the bulk to boundary. There has been an extensive use of software tools which include SageMath, Julia, to an extent SymPy for various calculations.

Acknowledgements

I would like to thank my mentor, Prof. Ling-Yan Hung, for the wonderful guidance, inspiration, enthusiam and valuable insights all along the length of the thesis. I would also like to thank her for giving me an opportunity to pursue my thesis at Fudan University. I would also like to thank my on-campus supervisior Dr. Prasanna Kumar, for all the inspiration and support throughout the course.

Thanks to my family and friends for the constant support.

Contents

List of Figures

Abbreviations

UTC	**U**nitary **T**ensor **C**ategory
MTC	**M**odular **T**ensor **C**ategory
FC	**F**usion **C**atgory
RFC	**R**ibbon **F**usion **C**ategory
UMTC	**U**nitary **M**odular **T**ensor **C**ategory

Part I

Chapter 1

Mathematical Preliminaries

1.1 Category Theory : Definitions and Structures

The aim of the section is to provide an overview of the various definitions and formalisms which will be used later in the presentation. To begin with Linearity, Semisimplicity, Finiteness are defined and then move onto Monoidal Categories,the later sections assume Monoidal Categories as a base to introduce other structures like Braiding, Rigidity and Twist, leading to the definition of Modularity, leading to Modular Tensor Categories.

1.1.1 Linearity, Semisimplicity, Finiteness

Definition 1.1. Linearity :
A category C is said to be linear, if the Homset(A,B) i.e., the set of morphisms from A to B, $\forall$ A,B $\in$ C forms a vector space over a field of charecteristic zero.

Definition 1.2. Simple :
An object $x \in$ a category C is said to be simple if $Hom(x, x) \cong C$ (the set of complex numbers). That is the endomorphisms of x is equivalent to C.
For further insight refer [1].

Definition 1.3. Semisimplicity :
A category C is said to be semisimple, if every object in C can be written as a direct sum of simple objects in C.

Definition 1.4. Finite :

A category C is said to be finite, if the number of simple objects in C is finite.

1.1.2 Modular Tensor Categories

Definition 1.5. Monoidal Category:

A category M is said to be monoidal if it is equipped with the following structure :

1. A functor called the tensor product $\otimes : M \times M \longrightarrow M$ where $\otimes (x, y) = x \otimes y$ and
   $\otimes (f, g) = f \otimes g \; \forall$ objects x,$y \in M$ and $\forall$ morphisms f,g in M

2. Natural Isomorphisms called the assosciator:

$$\alpha_{x,y,z} : (x \otimes y) \otimes z \longrightarrow x \otimes (y \otimes z)$$

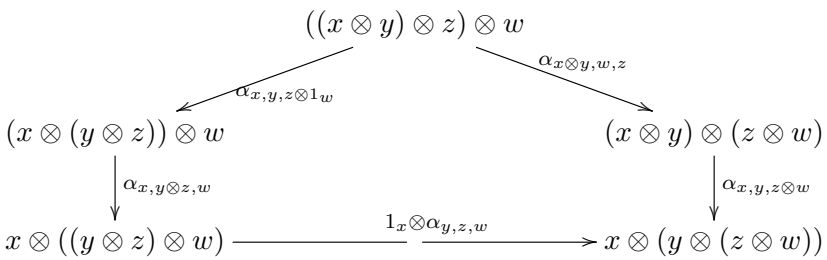
   The left unitor:

$$\beta_x : 1 \otimes x \longrightarrow x$$

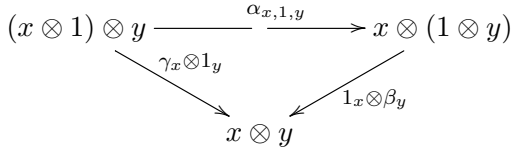
   The right unitor:

$$\gamma_x : x \otimes 1 \longrightarrow x$$

   such that the following diagrams commute.



   and



Definition 1.6. Rigidity :

A monoidal category M is said to be rigid, if for each object $A \in M$, $\exists$ an object $A^* \in M$ together with the maps :

$$i_A : 1 \to A \otimes A^*,$$
$$e_A : A^* \otimes A \to 1$$

Definition 1.7. Fusion Category :

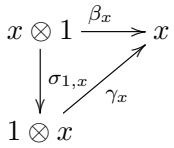
Fusion category is a finite semisimple C-linear rigid monoidal category such that monoidal unit is simple.

Definition 1.8. Braided Fusion Category :

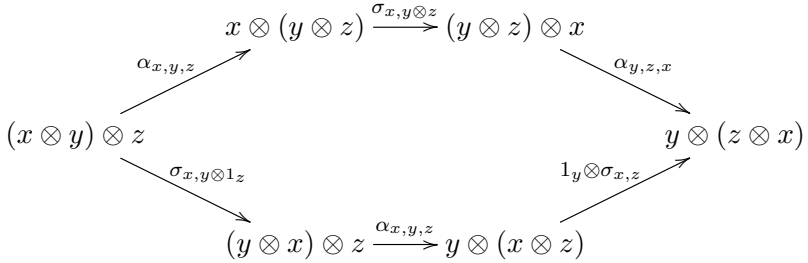
A braided fusion category M is a fusion category along with the following isomorphisms

$$\sigma_{x,y} : x \otimes y \longrightarrow \ y \otimes x$$

where x, $y \in M$ such that the following diagrams commute.



and



Definition 1.9. Pivotial Structure on a fusion category is an isomorphism,

$$\delta_A : A \to A^{**} \text{ ,such that}$$
$$\delta_{A\otimes B} = \delta_A \otimes \delta_B$$
$$\delta_1 = 1$$

Definition 1.10. Ribbon Fusion Category is a Braided Fusion Category with a pivotal structure which is compatible with braiding.

$$\theta_A = \gamma_A \circ \delta_A : A \to A \text{ via } A^{**}$$

Note : Braided Fusion Category with a pivotal structure is not always a Ribbon Fusion Category, as the pivtoal structure must be compatible with braiding.

Definition 1.11. Spherical Fusion Category :
Let $f \in End(X)$ where X is a simple object in C i.e., $f : X \longrightarrow X$ then right trace of the map f is given by

$$Tr^r(f) = 1 \to x \otimes x^* \to x^{**} \otimes x^* \to x \otimes x^* \to 1$$

On similar lines,

$$Tr^l(f) = 1 \to x^* \otimes x \to x^* \otimes x^{**} \to x^* \otimes x \to 1$$

Fusion Category is said to be spherical if $Tr^l(f) = Tr^r(f)\ \forall\ f \in$ C

Definition 1.12. Quantum Dimension of an object X in category C is given by $d_X := Tr(1_X)$, we define $D = \Sigma\ d_i^2$

Definition 1.13. We define the modular matrix S, whose elements are given by

$$s_{ij} = Tr(1_{i\otimes j}) = d_{i\otimes j},\ \forall\ i, j \in \text{C}$$

Finally we define Modular Tensor Category (MTC),

Definition 1.14. Modular Tensor Category :
MTC is a RFC such that the $det(S) \neq 0$.

1.1.3 Left, Right Modules over a Category

We define a left module over a tensor category, this is later used to define the boundaries for the Levin-Wen Models in Chapter 3.

Definition 1.15. Left Module Category:
Left Module Category over a monoidal category C, is a category M equipped with a C action : a functor $\otimes : C \otimes M \to M$ such that there are isomorphisms :

$$X \otimes (Y \otimes M') \to (X \otimes Y) \otimes M'$$
$$\mathbf{1} \otimes M' \to M'$$

for $X, Y \in C$ and $M' \in M$ satisfying some coherence conditions.

The definition of Right module is on similar lines.

1.2 Drinfeld Double of a Group

To introduce the Drinfeld Double of a Group, denoted by $D(G)$, we first define Algebras, Co-algebras, Bialgebras and Hopf Algebras.

1.2.1 Algebras, Co-algebras, Hopf Algebras

Definition 1.16. Algebra :
Let A be a vector space over a field K. The triple (A, m, η) is an associative algebra, where

$m : A \otimes A \to A$ (multiplication map),
$\eta : K \to A$ (unit map),
such that m, η satisfy the following commutation diagrams

$$\begin{array}{ccc} A \otimes A \otimes A & \xrightarrow{m \circ id_A} & A \otimes A \\ \downarrow{\scriptstyle id_A \circ m} & & \downarrow{\scriptstyle m} \\ A \otimes A & \xrightarrow{m} & A \end{array}$$

and

$$\begin{array}{ccc} A & \xrightarrow{id_A \circ \eta} & A \otimes A \\ \downarrow{\scriptstyle \eta \circ A} & & \downarrow{\scriptstyle m} \\ A \otimes A & \xrightarrow{m} & A \end{array}$$

Definition 1.17. Co-Algebra :

Let C be a vector space over a field K. The triple (C, n, δ) is a co-algebra, where

$n : C \to C \otimes C$ (comultiplication map)

$\delta : C \to K$ (counit map),

such that m, δ satisfy the following commutation diagrams

$$\begin{array}{ccc} C & \xrightarrow{n} & C \otimes C \\ \downarrow{\scriptstyle n} & & \downarrow{\scriptstyle id_C \circ n} \\ C \otimes C & \xrightarrow{n \circ id_C} & C \otimes C \otimes C \end{array}$$

and

$$\begin{array}{ccc} C & \xrightarrow{\delta} & C \otimes C \\ \downarrow{\scriptstyle \delta} & & \downarrow{\scriptstyle id_C \otimes \delta} \\ C \otimes C & \xrightarrow{\delta \otimes id_C} & C \end{array}$$

Definition 1.18. Bialgebra :

Let B be a vector space over a field K. A quintuple (B, m, n, η, δ) is a bialgebra, where (B, m, η) is an algebra and (B, n, δ) is a co-algebra.

Definition 1.19. Hopf Algebra:

Let H be a vector space over a field K. The quintuple (H, m, n, η, δ) along with the antipode map S

$$S : H \to H$$

such that the following diagram commutes

$$\begin{array}{ccccc} H \otimes H & & \xrightarrow{id_H \otimes S} & & H \otimes H \\ \uparrow{\scriptstyle n} & & & & \downarrow{\scriptstyle m} \\ H & \xrightarrow{\delta} & K & \xrightarrow{\eta} & H \\ \downarrow{\scriptstyle n} & & & & \uparrow{\scriptstyle m} \\ H \otimes H & & \xrightarrow{S \otimes id_H} & & H \otimes H \end{array}$$

forms a Hopf Algebra.

1.2.2 Drinfeld Double (Quantum Double) of a finite group

Let G be a finite group and K be a field. The group algebra $K[G]$ is the set of all linear combinations of elements from G with the scalars coming from the field K. This forms a vector space and by defining the comultiplication $n(g) := g \otimes g$ and counit by $\delta(g) = 1$, and the antipode $S(g) = g^{-1}$ turns the group algebra $K[G]$ into a Hopf Algebra by the above definitions.

Let $K(G)$ be the set of functions on G with values in K. The basis space is given by δ_g defined by the projection $\delta_g(\mathrm{h}) = \delta_{g,h}$. This forms a vector space and is an algebra with point-wise multiplication, with the unit given by $\eta : K \to K(G)$ is defined by $\eta(\lambda)(g) = \lambda$, with comultiplication, counit and antipode given by :

$$(nf)(g,h) = f(gh), \delta(f) = f(e), (Sf)(g) = f(g^{-1})$$

Thus, $K(G)$ is a Hopf Algebra using the above definitions.

Definition 1.20. Quantum Double of a group G, $D(G)$:
The vector space $K(G) \otimes K[G]$ along with the following structure :

$$\begin{gathered}(\delta_g \otimes x)(\delta_h \otimes y) = \delta_{gx,xh}(\delta_g \otimes xy),\\ 1 = \textstyle\sum_{g\in G} \delta_g \otimes e,\\ n(\delta_g \otimes x) = \textstyle\sum_{g_1 g_2 = g}(\delta_{g_1} \otimes x) \otimes (\delta_{g_2} \otimes x),\\ \Delta(\delta_g \otimes x) = \delta_{g,e},\\ S(\delta_g \otimes x) = \delta_{x^{-1}g^{-1}x} \otimes x^{-1}.\end{gathered}$$

forms a Hopf Algebra, which is defined as Quantum Doulbe of a group G, $D(G)$.

1.2.3 Representations of Drinfeld Double of a Group

Consider an element $a \in G$ and let π be a representation of $Z(a)$ over the vector space W with basis $\{w_1, ..., w_d\}$. Define the vector space $V_{\bar{a},\pi}$ with the basis $\{|b, w_i> : b \in$

$\bar{a}, 1 \leq i \leq d\}$. $V_{\bar{a},\pi}$ is a representation of $D(G)$ as follows. For any $b \in a$ fix $k_b \in G$ such that $b = k_b a k_b^{-1}$. Observe that $k_{gbg^{-1}}^{-1} g k_b$ is always in $Z(a)$, for any $w \in W$, $b \in \bar{a}$, and $gh^* \in D(G)$ define

$$gh^*|b, w> = \delta_{h,b}|gbg^{-1}, \pi(k_{gbg^{-1}}^{-1} g k_b)w>$$

The above action gives a representation of $D(G)$, the character of this representation is given by,

$$\chi_{(\bar{a},\pi)}(gh^*) = \delta_{h\in\bar{a}}\delta_{gh,hg} tr_\pi(k_h^{-1} g k_h)$$

All the irreducible representations of the $D(G)$ are indexed by the irreducible representations of the centralizer of the conjugacy classes. For more detailed treatment refer to [2, 3]

1.3 Algebras, Left (Right), BiModules over Algebras in a Category

The section aims to present the definitions of Algebras in a Category, Left(Right), Bi-Modules over Algebras in a category. These would be used later to outline the boundary excitations as presented by Kong [4] :

Definition 1.21. Algebra :
Let A be an object in a category C. An algebra is a triple (A, m, η) where

$m : A \otimes A \to A$, $\eta : K \to A$, where K is a simple object.
such that m, η satisfy the following commutation diagrams

$$\begin{array}{ccc} A\otimes A\otimes A & \xrightarrow{m\circ id_A} & A\otimes A \\ \downarrow{\scriptstyle id_A\circ m} & & \downarrow{\scriptstyle m} \\ A\otimes A & \xrightarrow{m} & A \end{array} \qquad \begin{array}{ccc} A & \xrightarrow{id_A\circ\eta} & A\otimes A \\ \downarrow{\scriptstyle \eta\circ A} & & \downarrow{\scriptstyle m} \\ A\otimes A & \xrightarrow{m} & A \end{array}$$

Given algebra A in category C, the right A-module is given by the following :

Definition 1.22. Right A-module :

The right module of an algebra A has objects as pairs (M, ρ_M) where $M \in C$, and ρ_M is given by $\rho_M : M \otimes A \to M$, such that the following commutation diagram is satisfied:

$$\begin{array}{ccc} M \otimes A \otimes A & \xrightarrow{\rho_M \circ id_A} & M \otimes A \\ \downarrow{\scriptstyle id_M \circ m} & & \downarrow{\scriptstyle \rho_M} \\ M \otimes A & \xrightarrow{\rho_M} & A \end{array}$$

Similarly we define the left module, a A-bimodule M is a triple equipped with both ρ^l_M and ρ^r_M.

Definition 1.23. Commutative Algebra:

Algebra (A, m, η) in C is said to be commutative if there exists a natural transformation $C_{M,A}$ given by

$$C_{M,A} : M \otimes A \to A \otimes M, \text{ where } M \in C$$

Definition 1.24. Separable Algebra :

Algebra (A, m, η) in C is called separable if there exists a bimodule map $e : A \to A \otimes A$ such that $m \circ e = id_A$. A separable algebra is called connected if $dim(hom(1, A)) = 1$)

Definition 1.25. Local Module over a commutative Algebra:

Let A be a commutative Algebra in C. Let (M, μ_M) be a right A-module. (M, μ_M) is called local if the following commutation diagram holds :

$$\begin{array}{ccc} A \otimes M & \xrightarrow{\mu_M} & M \\ \downarrow{\scriptstyle C_{A,M}} & & \uparrow{\scriptstyle \mu_M} \\ M \otimes A & \xrightarrow{C_{M,A}} & A \otimes M \end{array}$$

Chapter 2

Kitaev Quantum Double Model

2.1 Introduction to Quantum Double Models

This section is heavily inspired by [2]. Given a group G, consider a lattice with each edge being associated with a Hilbert space and indexed by a group element. A site in the lattice is given by a pair of adjacent vertex and face. For a given site (vertex v and face f), define the vertex operator A_v^g and face operator B_s^h as in figure 2.1 :

. The Hamiltonain for the lattice is given by

$$H = -\Sigma_v A_v - \Sigma_f B_f$$

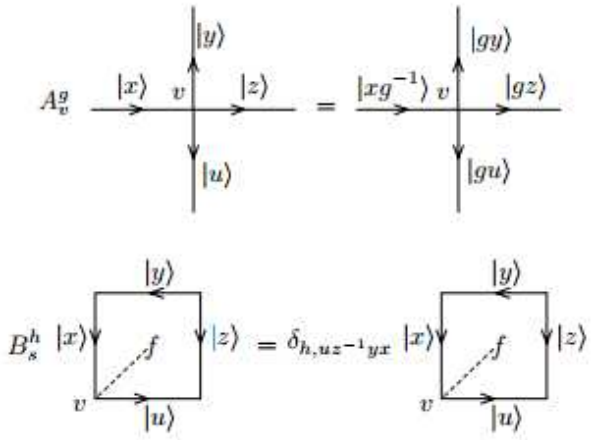


FIGURE 2.1: Definition of the A_v^g and B_s^h operators on a arbitrary vertex and face

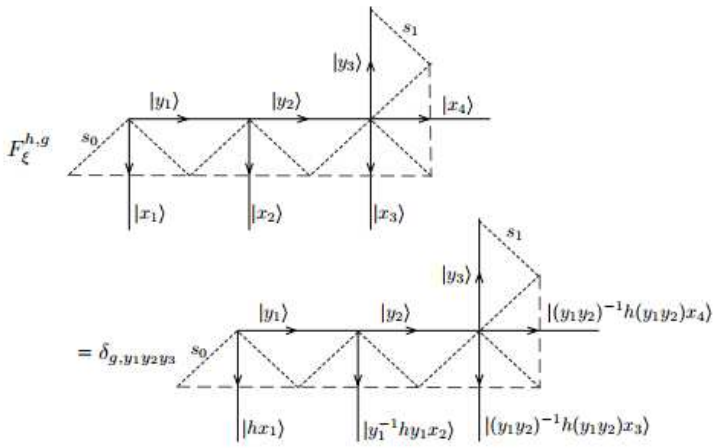


FIGURE 2.2: Definition of a ribbon operator on a arbitrary lattice

2.1.1 Ribbon operators, Excitations, Anyon types

A ribbion ξ in the lattice is a sequence of adjacent sites connecting two sites s_0 and s_1. The ribbon operator $F_\xi^{h,g}$ is defined as in figure 2.2 :

For a ribbon connecting the sites s_0 and s_1, and if any site in the adjacent sequence is given by t, the ribbon operator defined above satisfy the following commutation relationships,

$$
\begin{aligned}
[F_\xi^{h,g},\, A_t^k] = [F_\xi^{h,g},\, B_t^s] = 0,\\
A_{s_0}^k\ F_\xi^{h,g} = F_\xi^{khk^{-1},kg}\ A_{s_0}^k\\
B_{s_0}^k\ F_\xi^{h,g} = F_\xi^{h,g}\ B_{s_0}^{kh}\\
A_{s_1}^k\ F_\xi^{h,g} = F_\xi^{h,gk^{-1}}\ A_{s_1}^k\\
B_{s_1}^k\ F_\xi^{h,g} = F_\xi^{h,g}\ B_{s_1}^{g^{-1}h^{-1}gk}
\end{aligned}
$$

The application of ribbon operator on the ground state, gives rise to quasi-particle excitations at the end of the ribbon. The excitations are independent of the topology of the ribbon operator. Therefore the excitations can be moved around the lattice by extending/contracting the ribbon. Fusion of two quasi-particle excitations can be achieved by moving them to the same site and fusing them, the resultant describes a system of anyons.

Anyon types for a Kitaev Quantum Double are in one-to-one correspondance with the irreducdible representations of the Drinfeld Double of the group, which are in one-to-one

correspondace with the irreducible representations of the centralizers of the conjugacy classes of the group [2]. The method to compute this for any finite group has been presented in Appendix A and has been computed for various groups like Z_2, S_3, D_4 the first one being the case of Toric Code.

2.1.2 Introduction of boundaries, Condensates, Ribbon operators

Consider a lattice as above but with a boundary, that is with a lattice on one half and nothing on the other side, and the edges connecting both the half planes being the boundary. The edges on the boundary are associated with a Hilbert space $C[K]$, where C is the complex field and $K \subset G$ and indexed by elements of K. The vertex and the face operator for the internal lattice remain as defined in the introduction, but for the boundary the vertex and face operators are defined as follows :

$$A_s^K = \frac{1}{|K|} \Sigma_{k \in K} A_s^k$$
$$B_s^K = \Sigma_{k \in K} B_s^k$$

where s is a site on the boundary.

The hamiltonian of the system with the boundary is given by,

$$H = -\Sigma_v A_v - \Sigma_f B_f - \Sigma_s (A_s^K + B_s^K)$$

To construct a ribbon $T = \Sigma_{h,g} c_{h,g} F_\xi^{h,g}$, connecting the sites in the bulk to the sites on the boundary, the ribbon should satisfy the commutation relationship mentioned in the previous section

$$A_{s_0}^k \; T = T \; A_{s_0}^k$$
$$B_{s_0}^k \; T = F_\xi^{h,g} \; B_{s_0}^{kh}$$

along with the following commutation relationships :

$$[T, A_{s_0}^K] = 0 \; ,$$
$$[T, B_{s_0}^K] = 0$$

Solving for T, gives

$$T^{(k,g)} = \Sigma_{l \in K} F_{\xi}^{(lkl^{-1}, lg^{-1})}$$

We now present the various $T^{(k,g)}$ for the subgroups of S_3. Consider the subgroup $K = G$, we compute the $T^{(k,g)}$ for various combinations of k and g.

2.1.2.1 Ribbon operators in a lattice with boundaries in the case of S_3

Using the following snippet, we group the summation for each of the subgroup

```
sage: G =  SymmetricGroup(3)
sage: K = G.subgroups()[5]
sage: x = []
sage: y = []
sage: for k in K:
    for g in G:
        for l in K:
            x.append([l*k*l^-1, l*g^-1])
....:
sage: for i in range(len(x)/6):
    y.append(x[6*i:6*(i+1)])

sage: y
# The first element is given by lkl^-1 and second element by
    lg^-1, which allows
# us to read  of the T^{(k,g)} from the first element of
   every summation.

# For the case k = e and g running over all group elements.

[[[(), ()], [(), (2,3)], [(), (1,2,3)], [(), (1,2)],
[(), (1,3,2)], [(), (1,3)]],
[[(), (1,2)], [(), (1,2,3)], [(), (2,3)], [(), ()],
[(), (1,3)], [(), (1,3,2)]],
[[(), (1,3,2)], [(), (1,3)], [(), ()], [(), (2,3)],
[(), (1,2,3)], [(), (1,2)]],
```

```
[[(), (1,2,3)], [(), (1,2)], [(), (1,3,2)], [(), (1,3)],
[(), ()], [(), (2,3)]],
[[(), (2,3)], [(), ()], [(), (1,3)], [(), (1,3,2)],
[(), (1,2)], [(), (1,2,3)]],
[[(), (1,3)], [(), (1,3,2)], [(), (1,2)], [(), (1,2,3)],
[(), (2,3)], [(), ()]],

# Therefore, T^{(e,e)} is given by the first sum and
   similarly others.
# So for the element e, the operators collapse to a single
   operator given by
# T^{(e,g)} = summation(F^{(e,g)}) over all g \in G.
   Implying all the six collapse to a single operator.

# For the case k = (2,3) and g running over all group
   elements.

[[(2,3), ()], [(2,3), (2,3)], [(1,2), (1,2,3)], [(1,3),
   (1,2)],
[(1,3), (1,3,2)], [(1,2), (1,3)]],
[[(2,3), (1,2)], [(2,3), (1,2,3)], [(1,2), (2,3)], [(1,3),
   ()],
[(1,3), (1,3)], [(1,2), (1,3,2)]],
[[(2,3), (1,3,2)], [(2,3), (1,3)], [(1,2), ()], [(1,3),
   (2,3)],
[(1,3), (1,2,3)], [(1,2), (1,2)]],
[[(2,3), (1,2,3)], [(2,3), (1,2)], [(1,2), (1,3,2)], [(1,3),
    (1,3)],
[(1,3), ()], [(1,2), (2,3)]],
[[(2,3), (2,3)], [(2,3), ()], [(1,2), (1,3)], [(1,3),
   (1,3,2)],
[(1,3), (1,2)], [(1,2), (1,2,3)]],
```

```
[[(2,3), (1,3)], [(2,3), (1,3,2)], [(1,2), (1,2)], [(1,3),
   (1,2,3)],
[(1,3), (2,3)], [(1,2), ()]],

# In this case, (Performing a right multiplication gives the
    same results !)
# T^{((2,3), e)} = T^{((2,3), (2,3))},
# T^{((2,3), (1,2))} = T^{((2,3), (1,3,2))}
# T^{((2,3), (1,2,3))} = T^{((2,3), (1,3))}

# Therefore 6 operators reduce to 3.

# For the case k = (1,2,3) and g running over all group
   elements.

[[(1,2,3), ()], [(1,3,2), (2,3)], [(1,2,3), (1,2,3)],
   [(1,3,2), (1,2)],
[(1,2,3), (1,3,2)], [(1,3,2), (1,3)]],
[[(1,2,3), (1,2)], [(1,3,2), (1,2,3)], [(1,2,3), (2,3)],
   [(1,3,2), ()],
[(1,2,3), (1,3)], [(1,3,2), (1,3,2)]],
[[(1,2,3), (1,3,2)], [(1,3,2), (1,3)], [(1,2,3), ()],
   [(1,3,2), (2,3)],
[(1,2,3), (1,2,3)], [(1,3,2), (1,2)]],
[[(1,2,3), (1,2,3)], [(1,3,2), (1,2)], [(1,2,3), (1,3,2)],
   [(1,3,2), (1,3)],
[(1,2,3), ()], [(1,3,2), (2,3)]],
[[(1,2,3), (2,3)], [(1,3,2), ()], [(1,2,3), (1,3)], [(1,3,2)
   , (1,3,2)],
[(1,2,3), (1,2)], [(1,3,2), (1,2,3)]],
[[(1,2,3), (1,3)], [(1,3,2), (1,3,2)], [(1,2,3), (1,2)],
   [(1,3,2), (1,2,3)],
[(1,2,3), (2,3)], [(1,3,2), ()]],
```

```
# In this case, again using the same rule as above it is
   easy to see
# T^{((1,2,3), e)} = T^{((1,2,3), (1,2,3))} = T^{((1,2,3),
   (1,3,2))}
# T^{((1,2,3), (1,2))} = T^{((1,2,3), (1,3))} = T^{((1,2,3),
    (2,3))}

# Therefore 6 operators collapse to 2.

# For the case k = (1,2) and g running over all group
   elements.

[[(1,2), ()], [(1,3), (2,3)], [(1,3), (1,2,3)], [(1,2),
   (1,2)],
[(2,3), (1,3,2)], [(2,3), (1,3)]],
[[(1,2), (1,2)], [(1,3), (1,2,3)], [(1,3), (2,3)], [(1,2),
   ()],
[(2,3), (1,3)], [(2,3), (1,3,2)]],
[[(1,2), (1,3,2)], [(1,3), (1,3)], [(1,3), ()], [(1,2),
   (2,3)],
[(2,3), (1,2,3)], [(2,3), (1,2)]],
[[(1,2), (1,2,3)], [(1,3), (1,2)], [(1,3), (1,3,2)], [(1,2),
    (1,3)],
[(2,3), ()], [(2,3), (2,3)]],
[[(1,2), (2,3)], [(1,3), ()], [(1,3), (1,3)], [(1,2),
   (1,3,2)],
[(2,3), (1,2)], [(2,3), (1,2,3)]],
[[(1,2), (1,3)], [(1,3), (1,3,2)], [(1,3), (1,2)], [(1,2),
   (1,2,3)],
[(2,3), (2,3)], [(2,3), ()]]
```

```
# In this case, again using the same rule as above it is
   easy to see
# T^{((1,2), e)} = T^{((1,2), (1,2))},
# T^{((1,2), (1,3))} = T^{((1,2), (1,3,2))}
# T^{((1,2), (1,2,3))} = T^{((1,2), (2,3))}

# But in this case we also have
# T^{((1,2), e)} = T^{((2,3), (1,2,3))}
# T^{((1,2), (1,3))} = T^{((2,3), e)}
# T^{((1,2), (1,2,3))} = T^{((2,3), (1,3,2))

# So again in this case 6  operators reduce to 3 but these
   are  mapped to the previous maps.

# For the case k = (1,3,2) and g running over all group
   elements.

[[(1,3,2), ()], [(1,2,3), (2,3)], [(1,3,2), (1,2,3)],
   [(1,2,3), (1,2)],
[(1,3,2), (1,3,2)], [(1,2,3), (1,3)]],
[[(1,3,2), (1,2)], [(1,2,3), (1,2,3)], [(1,3,2), (2,3)],
   [(1,2,3), ()],
[(1,3,2), (1,3)], [(1,2,3), (1,3,2)]],
[[(1,3,2), (1,3,2)], [(1,2,3), (1,3)], [(1,3,2), ()],
   [(1,2,3), (2,3)],
[(1,3,2), (1,2,3)], [(1,2,3), (1,2)]],
[[(1,3,2), (1,2,3)], [(1,2,3), (1,2)], [(1,3,2), (1,3,2)],
   [(1,2,3), (1,3)],
[(1,3,2), ()], [(1,2,3), (2,3)]],
[[(1,3,2), (2,3)], [(1,2,3), ()], [(1,3,2), (1,3)], [(1,2,3)
   , (1,3,2)],
[(1,3,2), (1,2)], [(1,2,3), (1,2,3)]],
```

```
[[(1,3,2), (1,3)], [(1,2,3), (1,3,2)], [(1,3,2), (1,2)],
   [(1,2,3), (1,2,3)],
[(1,3,2), (2,3)], [(1,2,3), ()]],

# In this case, again using the same rule as above it is
   easy to see
# T^{((1,3,2), e)} = T^{((1,3,2), (1,3,2))} = T^{((1,3,2),
   (1,2,3))}
# T^{((1,2,3), (1,2))} = T^{((1,2,3), (1,3))} = T^{((1,2,3),
    (2,3))}

# But in this case we also have
# T^{((1,3,2), e)} = T^{((1,2,3), (1,2))}
# T^{((1,3,2), (1,2))} = T^{((1,2,3), e)}

# As expected we have a reduction from 6 to 2, but these are
     again mapped.

# For the case k = (1,2) and g running over all group
   elements.
[[(1,3), ()], [(1,2), (2,3)], [(2,3), (1,2,3)], [(2,3),
   (1,2)],
[(1,2), (1,3,2)], [(1,3), (1,3)]],
[[(1,3), (1,2)], [(1,2), (1,2,3)], [(2,3), (2,3)], [(2,3),
   ()],
[(1,2), (1,3)], [(1,3), (1,3,2)]],
[[(1,3), (1,3,2)], [(1,2), (1,3)], [(2,3), ()], [(2,3),
   (2,3)],
[(1,2), (1,2,3)], [(1,3), (1,2)]],
[[(1,3), (1,2,3)], [(1,2), (1,2)], [(2,3), (1,3,2)], [(2,3),
    (1,3)],
[(1,2), ()], [(1,3), (2,3)]],
```

```
[[(1,3), (2,3)], [(1,2), ()], [(2,3), (1,3)], [(2,3),
   (1,3,2)],
[(1,2), (1,2)], [(1,3), (1,2,3)]],
[[(1,3), (1,3)], [(1,2), (1,3,2)], [(2,3), (1,2)], [(2,3),
   (1,2,3)],
[(1,2), (2,3)], [(1,3), ()]]]

# In this case, again using the same rule as above it is
   easy to see
# T^{((1,3), e)} = T^{((1,3), (1,3))},
# T^{((1,3), (1,2,3))} = T^{((1,3), (1,2))}
# T^{((1,3), (1,3,2))} = T^{((1,3), (2,3))}

# But in this case we also have
# T^{((1,3), e)} = T^{((1,2), (2,3))}
# T^{((1,3), (1,2,3))} = T^{((1,2), (1,3,2)}
# T^{((1,3), (1,3,2))} = T^{((1,2), (1,2))

# So again in this case 6  operators reduce to 3 but these
   are  mapped to the previous maps.
```

So in the above case where $K = G$ we have reduced 36 operators to 6 unique operators.

Carrying out a similar analysis for $K = \{e\}$, we again end up with 6 operators as follows :

```
sage: G = SymmetricGroup(3)
sage: K = G.subgroups()[0]
sage: K
Subgroup of (Symmetric group of order 3! as a permutation
   group) generated by [()]
sage: for k in K:
    for g in G:
        for l in K:
            print k,g,l*k*l^-1, l*g^-1
```

```
....:
() () () ()
() (1,2) () (1,2)
() (1,2,3) () (1,3,2)
() (1,3,2) () (1,2,3)
() (2,3) () (2,3)
() (1,3) () (1,3)
```

Hence the unique 6 operators are given by $T^{(e,e)}$, $T^{(e,(1,2))}$, $T^{(e,(2,3))}$, $T^{(e,(1,3))}$, $T^{(e,(1,2,3))}$, $T^{(e,(1,3,2))}$ give out $F^{(e,e)}$, $F^{(e,(1,2))}$, $F^{(e,(2,3))}$, $F^{(e,(1,3))}$, $F^{(e,(1,2,3))}$, $F^{(e,(1,3,2))}$ in terms of $F_{\xi}^{(h,g)}$ which is defined on the sites.

Carrying out a similar analysis for $K = \{e, \tau\}$, we again end up with 6 operators as follows :

```
sage: G = SymmetricGroup(3)
sage: K = G.subgroups()[1]
sage: K
Subgroup of (Symmetric group of order 3! as a permutation
   group) generated by [(2,3)]
sage: for k in K:
    for g in G:
        for l in K:
            print k,g,l*k*l^-1, l*g^-1
() () () ()
() () () (2,3)
() (1,2) () (1,2)
() (1,2) () (1,2,3)
() (1,2,3) () (1,3,2)
() (1,2,3) () (1,3)
() (1,3,2) () (1,2,3)
() (1,3,2) () (1,2)
() (2,3) () (2,3)
() (2,3) () ()
() (1,3) () (1,3)
```

```
() (1,3) () (1,3,2)
(2,3) () (2,3) ()
(2,3) () (2,3) (2,3)
(2,3) (1,2) (2,3) (1,2)
(2,3) (1,2) (2,3) (1,2,3)
(2,3) (1,2,3) (2,3) (1,3,2)
(2,3) (1,2,3) (2,3) (1,3)
(2,3) (1,3,2) (2,3) (1,2,3)
(2,3) (1,3,2) (2,3) (1,2)
(2,3) (2,3) (2,3) (2,3)
(2,3) (2,3) (2,3) ()
(2,3) (1,3) (2,3) (1,3)
(2,3) (1,3) (2,3) (1,3,2)
```

It is easy to see that the 12 operators reduce to unique 6,
$T^{(e,e)} = F^{(e,e)} + F^{(e,(2,3))}$,
$T^{(e,(1,2))} = F^{(e,(1,2))} + F^{(e,(1,2,3))}$,
$T^{(e,(1,2,3))} = F^{(e,1,3,2)} + F^{(e,(1,3))}$,
$T^{((2,3),e)} = F^{((2,3),e)} + F^{((2,3),(2,3))}$,
$T^{((2,3),(1,2))} = F^{((2,3),(1,2))} + F^{((2,3),(1,2,3))}$,
$T^{((2,3),(1,2,3)} = F^{((2,3),(1,3,2))} + F^{((2,3),(1,3))}$,

2.1.2.2 Explicit Ribbon operators with boundary $K = \{e, \tau\} \subset S_3$ in terms of basis

The section aims to present the explicit form of the operators generated in the case of $K = \{e, \tau\}$, in terms of the basis of the algebra of the ribbon operators [5]. The F operators are represented in terms of the basis $F^{RC}u(i,j)v(i',j')$, where R is the irreducible representation of the center of the conjugacy class C, is given by:

$$F_{\xi}^{h,g} = \Sigma_{R \in N_{C irred}} \Sigma_{j,j'=1}^{n_R} \Gamma_R^{j,j'}(n_{(h,g)}) F^{RC} u(i,j) v(i',j')$$

where $h^{-1} \in C$, $n_{(h,g)} = q^{-1}_{i(h^{-1})} g q_{i(g^{-1}h^{-1}g)}$ Here Γ is the unitary matrix representation of the element $n_{(h,g)}$. We list the representation of S_3 as it is the center of the conjugacy class of e.

Elements	1-dim	1-dim	2-dim
e	[1]	[1]	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
$(1,2,3)$	[1]	[1]	$\begin{bmatrix} -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}$
$(1,3,2)$	[1]	[1]	$\begin{bmatrix} -\frac{1}{2} & -\frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}$
$(1,2)$	[1]	[-1]	$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$
$(2,3)$	[1]	[-1]	$\begin{bmatrix} \frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}$
$(1,3)$	[1]	[-1]	$\begin{bmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}$

$F^{(e,e)}$, here h^{-1} is in the conjugacy class of e and its centralizer is S_3 itself. So we have three irreps π_1, π_2, π_3 whose degrees are 1,1,2. Therefore $F^{(e,e)}$ is given by :

$$\begin{aligned}\Gamma^{1,1}_{\pi_1}(n_{(e,e)})F^{\pi_1\bar{e}}u(i,1)v(i',1) + \Gamma^{1,1}_{\pi_2}(n_{(e,e)})F^{\pi_2\bar{e}}u(i,1)v(i',1) + \\ \Gamma^{1,1}_{\pi_3}(n_{(e,e)})F^{\pi_3\bar{e}}u(i,1)v(i',1) + \Gamma^{1,2}_{\pi_3}(n_{(e,e)})F^{\pi_3\bar{e}}u(i,1)v(i',2) + \\ \Gamma^{2,1}_{\pi_3}(n_{(e,e)})F^{\pi_3\bar{e}}u(2,1)v(i',1) + \Gamma^{2,2}_{\pi_3}(n_{(e,e)})F^{\pi_3\bar{e}}u(i,2)v(i',2)\end{aligned}$$

which reduces to

$$F^{(e,e)} = F^{\pi_1\bar{e}}u(i,1)v(i',1) + F^{\pi_2\bar{e}}u(i,1)v(i',1) + F^{\pi_3\bar{e}}u(i,1)v(i',1) + F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

$F^{(e,(2,3))}$ is given by a similar structure, noting that $n_{(e,(2,3))} = (2,3)$, therefore the sum reduces to

$$\begin{aligned}F^{(e,(2,3))} = F^{\pi_1\bar{e}}u(i,1)v(i',1) - F^{\pi_2\bar{e}}u(i,1)v(i',1) + \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,1)v(i',1) + \\ \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,1)v(i',2) + \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,2)v(i',1) - \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,2)v(i',2)\end{aligned}$$

Therefore $T^{(e,e)} = F^{(e,e)} + F^{(e,(2,3))}$, is given by

$$T^{(e,e)} = 2F^{\pi_1\bar{e}}u(i,1)v(i',1) + \tfrac{3}{2}F^{\pi_3\bar{e}}u(i,1)v(i',1) + \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,1)v(i',2) + \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,2)v(i',1) + \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

On similar lines, $T^{(e,(1,2))} = F^{(e,(1,2))} + F^{(e,(1,2,3))}$ we need to compute $F^{(e,(1,2))}$ and $F^{(e,(1,2,3))}$

For $F^{(e,(1,2))}$, we need $n_{(e,(1,2))} = (1,2)$ and hence

$$F^{(e,(1,2))} = F^{\pi_1\bar{e}}u(i,1)v(i',1) - F^{\pi_2\bar{e}}u(i,1)v(i',1) - F^{\pi_3\bar{e}}u(i,1)v(i',1) + F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

For $F^{(e,(1,2,3))}$, we need $n_{(e,(1,2,3))} = (1,2,3)$ and hence

$$F^{(e,(1,2,3))} = F^{\pi_1\bar{e}}u(i,1)v(i',1) + F^{\pi_2\bar{e}}u(i,1)v(i',1) - \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,1)v(i',1) + \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,1)v(i',2) - \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,2)v(i',1) - \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

Therefore $T^{(e,(1,2))}$ is given by

$$T^{(e,(1,2))} = 2F^{\pi_1\bar{e}}u(i,1)v(i',1) - \tfrac{3}{2}F^{\pi_3\bar{e}}u(i,1)v(i',1) + \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,1)v(i',2) - \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,2)v(i',1) + \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

On similar lines, $T^{(e,(1,2,3))} = F^{(e,(1,3,2))} + F^{(e,(1,3))}$ we need to compute $F^{(e,(1,3,2))}$ and $F^{(e,(1,3))}$

For $F^{(e,(1,3,2))}$, we need $n_{(e,(1,3,2))} = (1,3,2)$ and hence

$$F^{(e,(1,3,2))} = F^{\pi_1\bar{e}}u(i,1)v(i',1) + F^{\pi_2\bar{e}}u(i,1)v(i',1) - \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,1)v(i',1) - \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,1)v(i',2) + \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,2)v(i',1) - \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

For $F^{(e,(1,3))}$, we need $n_{(e,(1,3))} = (1,3)$ and hence

$$F^{(e,(1,3))} = F^{\pi_1\bar{e}}u(i,1)v(i',1) - F^{\pi_2\bar{e}}u(i,1)v(i',1) + \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,1)v(i',1) - \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,1)v(i',2) - \tfrac{\sqrt{3}}{2}F^{\pi_3\bar{e}}u(i,2)v(i',1) - \tfrac{1}{2}F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

Therefore $T^{(e,(1,3,2))}$ is given by

$$T^{(e,(1,3,2))} = 2F^{\pi_1\bar{e}}u(i,1)v(i',1) - \sqrt{3}F^{\pi_3\bar{e}}u(i,1)v(i',2) - F^{\pi_3\bar{e}}u(i,2)v(i',2)$$

On similar lines, $T^{((2,3),e)} = F^{((2,3),e)} + F^{((2,3),(2,3))}$ we need to compute $F^{((2,3),e)}$ and $F^{((2,3),(2,3))}$

For $F^{((2,3),e)}$, we need $n_{((2,3),e)} = e$ and hence

$$F^{((2,3),e)} = F^{\pi_1\bar{\tau}}u(i,1)v(i',1) + F^{\pi_2\bar{\tau}}u(i,1)v(i',1)$$

For $F^{((2,3),(2,3))}$, we need $n_{((2,3),(2,3))} = (2,3)$ and hence

$$F^{(2,3),e} = F^{\pi_1\bar{\tau}}u(i,1)v(i',1) - F^{\pi_2\bar{\tau}}u(i,1)v(i',1)$$

Therefore $T^{(2,3),e)}$ is given by

$$T^{((2,3),e)} = 2F^{\pi_1\bar{\tau}}u(i,1)v(i',1)$$

Similarly we have the other two ribbon operators given by

$$T^{((2,3),(1,2))} = 2F^{\pi_1\bar{\tau}}u(i,1)v(i',1)$$

$$T^{((2,3),(1,2,3))} = 2F^{\pi_1\bar{\tau}}u(i,1)v(i',1)$$

Here the i' varies over 1,2,3, giving rise to different T operators.

The above representation was computed with an idea to further reduce the number of operators, with an expectation of compressing 6 operators to three which further give rise to three ground states (thus verifying the fact that there are three ground states in the given system). But, as seen this happens to be a basis transformation, the calculation of the ground state is presented in the next section.

2.1.2.3 Excitation condensation on various boundaries for $D(S_3)$

The excitations in a Quantum Double model are given by irreps of the centralizers of the conjugacy class of the group. The character of a particular representation π is given by $\chi_{(\bar{a},\pi)}$:

$$\chi_{(\bar{a},\pi)}(gh^*) = \delta_{h\in\bar{a}}\delta_{gh,hg}tr_\pi(k_h^{-1}gk_h),$$

as $h \in \bar{a}$ is one of the conditions to be satisfied, k_h is some element in G such that $h = k_h a k_h^{-1}$

For $k \in K$ and $g \in G$ define $|\psi_K^{k,g} >= T^{(k,g)}|\psi_K >$ and let $A(K)$ be the span of these vectors. Let $\chi_{A(K)}$ be the character of the representation $A(K)$.

$$\chi_{A(K)}(hg^*) = \frac{1}{|K|}\delta_{gh,hg}\Sigma_{x\in G}\delta_{xgx^{-1}\in K}\delta_{xhx^{-1}\in K}$$

To compute whether a particular excitation condenses at the boundary, the inner product defined as

$$< \chi_1, \chi_2 >= \frac{1}{|G|}\Sigma_{g,h}(\chi_1(gh^*)^*\chi_2(gh^*)$$

of the above characters is observed, if it is positive the excitation associated with the irrep of $D(G)$ condenses at the boundary.

Consider the case of S_3, the following are the conjugacy classes $\{e\}$, $\{\tau\}$ and $\{\sigma\}$ where $\{e, \tau, \sigma\}$ form the presentation of the group. The corresponding centralizer for each of the conjugacy classes are given by S_3, $\{e, \tau\}$, $\{e, \sigma, \sigma^{-1}\}$, whose corresponding irreps are $\{1$, sign, $\pi\}$, (where 1, sign are one dimensional irreps of S_3 and π is a two dimensional irrep.), $\{1, -1\}$, $\{1, \omega, \omega^*\}$. For the case of S_3 the irreps of $D(S_3)$ are indexed by $V_{a,\pi}$ where π is the representation of the centralizer of the conjugacy class of a. Put together there are eight excitations in $D(S_3)$ given by V_{e,π_1}, V_{e,π_2}, V_{e,π_3}, V_{τ,π_1}, V_{τ,π_2}, V_{σ,π_1}, V_{σ,π_2}, V_{sigma,π_3} which are labelled and referred by A, B, C, D, E, F, G, G. The following snippet has been used to compute the inner product,

```
sage: def character_1_irred(G, subgroup, g, h):
    sum = 0
    if h*g == g*h:
        for i in G:
            if i*g*i^-1 in subgroup and i*h*i^-1 in subgroup
   :
                sum = sum + 1
    return sum/len(subgroup)
....:
sage: def character_2_irred(G, conjugacy_class, g, h):
```

```
    k_h = 0
    for i in G:
        if h*i == i*conjugacy_class.an_element():
            k_h = i
            break
    if g*h == h*g and k_h != 0:
        return  k_h^-1*g*k_h
    else:
        return 0
```

In the above snippet the first function returns the irreducible character of $A(K)$. The second function returns the element on which the trace is to be calculated by relating to the representation of the centralizer of the conjugacy class.

S_3 is taken as an example to explain further. First the subgroups and the conjugacy classes of S_3 are presented.

```
sage: G = SymmetricGroup(3)
sage: for i in G.subgroups():
....:     print i
....:
Subgroup of (Symmetric group of order 3! as a permutation
   group)
generated by [()]
Subgroup of (Symmetric group of order 3! as a permutation
   group)
generated by [(2,3)]
Subgroup of (Symmetric group of order 3! as a permutation
   group)
generated by [(1,2)]
Subgroup of (Symmetric group of order 3! as a permutation
   group)
generated by [(1,3)]
Subgroup of (Symmetric group of order 3! as a permutation
   group)
```

```
generated by [(1,2,3)]
Subgroup of (Symmetric group of order 3! as a permutation
   group)
generated by [(2,3), (1,2,3)]

sage: for i in G.conjugacy_classes():
....:     print i
....:
Conjugacy class of cycle type [1, 1, 1] in
Symmetric group of order 3! as a permutation group
Conjugacy class of cycle type [2, 1] in
Symmetric group of order 3! as a permutation group
Conjugacy class of cycle type [3] in
Symmetric group of order 3! as a permutation group
```

Consider the subgroup K = G, and for each of the conjugacy classes the element on which the trace is to be calculated is presented :

```
sage: G.subgroups()[5]
Subgroup of (Symmetric group of order 3! as a permutation
   group)
generated by [(2,3), (1,2,3)]

sage: G.conjugacy_classes()[0]
Conjugacy class of cycle type [1, 1, 1] in
Symmetric group of order 3! as a permutation group

sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[5], i, j),
               character_2_irred(G, G.conjugacy_classes()
   [0], i, j))
....:
(1, ())
```

```
(1, (1,2))
(1, (1,2,3))
(1, (1,3,2))
(1, (2,3))
(1, (1,3))

sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[5], i, j),
               character_2_irred(G, G.conjugacy_classes()
   [1], i, j))
....:
(1, ())
(1, ())
(1, ())
(1, (1,2))
(1, (1,2))
(1, (1,2))

sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[5], i, j),
               character_2_irred(G, G.conjugacy_classes()
   [2], i, j))
....:
(1, ())
(1, ())
(1, (1,2,3))
(1, (1,3,2))
(1, (1,3,2))
(1, (1,2,3))
```

The reading of the list is as follows : the first loop reads the following :

$$1^{*}tr_{\pi_i}(e) + 1^{*}tr_{\pi_i}(1,2) + 1^{*}tr_{\pi_i}(2,3) + 1^{*}tr_{\pi_i}(1,3) + 1^{*}tr_{\pi_i}(1,2,3) + 1^{*}tr_{\pi_i}(1,3,2)$$

The character table of S_3 is given by, which gives the trace of the irreducible representations (here in this case the irreducible representation of the centralizer of the conjugacy class of e is same as the irreducible representations of group S_3) :

```
sage: G.character_table()
[ 1 -1  1] - $\pi_{2}$
[ 2  0 -1] - $\pi_{3}$
[ 1  1  1] - $\pi_{1}$
```

Let the respective representations be π_1, π_2, π_3. From the character table for the conjugacy class of e have V_{e,π_1} to be a condensate as the inner product is greater than zero. Considering the case of π_2 the inner product goes to zero therefore V_{e,π_2} does not condense at the boundary. Similarly the case of V_{e,π_3}.

Consider the conjugacy class of $\{\tau\}$, the centralizer of the conjugacy classes has irreducible representations as 1 and -1. Reading from the second loop we have the inner product as the following :

$$3^{*}tr_{\pi_i}(e) + 3^{*}tr_{\pi_i}(1,2)$$

which is greater than zero for 1 and zero for -1. Therefore V_{τ,π_1} forms a condensate while the other does not.

Similarly for the conjugacy class of $\{\sigma\}$, the centralizer of the conjugacy class has irreducible representations as 1, ω, and ω^*. The inner product is given by :

$$2^{*}tr_{\pi_i}(e) + 2^{*}tr_{\pi_i}(1,2,3) + 2^{*}tr_{\pi_i}(1,3,2)$$

Therefore V_{σ,π_1} condense while the other two do not.

Therefore to conclude, in the case of $K = G$, A,D,F condense at the boundary.

Now consider the subgroup K=$\{e\}$, we perform a similar analysis :

```
sage: for i in G:
....:     for j in G:
```

```
....:          print (character_1_irred(G, G.subgroups()[0],
   i, j),
                      character_2_irred(G, G.
   conjugacy_classes()[0], i, j))
....:
(6, ())
```

For the conjugacy class of e we end up with the following expression for the inner product :

$$6^{*}tr_{\pi_i}(e), \text{for each of } \pi_i$$

the inner product is observed to be greater than zero. Therefore V_{e,π_i}, for i = 1,2,3 condense at the boundary.

Therefore to conclude, in the case of $K = e$, A,B,C condense at the boundary.

For the other two conjugacy classes i.e., $\{\tau\}$ and $\{\sigma\}$ the inner product goes to zero, therefore none of the excitations indexed by this space condense at the boundary.

Consider the subgroup K=$\{e, \tau\}$, we have the following :

```
sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[1], i, j),
               character_2_irred(G, G.conjugacy_classes()
   [0], i, j))
....:
(3, ())
(1, (1,2))
(1, (2,3))
(1, (1,3))

sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[1], i, j),
```

```
                character_2_irred(G, G.conjugacy_classes()
   [1], i, j))
....:
(1, ())
(1, ())
(1, ())
(1, (1,2))
(1, (1,2))
(1, (1,2))

sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[1], i, j),
                character_2_irred(G, G.conjugacy_classes()
   [2], i, j))
....:
```

The above results imply the following : For the conjugacy class of e, the inner product results in the following:

$$3{*}tr_{\pi_i}(e) + 1{*}tr_{\pi_i}(1,2) + 1{*}tr_{\pi_i}(2,3) + 1{*}tr_{\pi_i}(1,3)$$

which results in a value greater than zero for i=1 and i=3. Therefore V_{e,π_1}, V_{e,π_3} condense at the boundary while V_{e,π_2} do not.

Looking through the second loop gives the following :

$$3{*}tr_{\pi_i}(e) + 3{*}tr_{\pi_i}(1,2)$$

which results in a value greater than zero for the trivial representation that is π_1 while the other goes to zero. Therefore V_{τ,π_1} condense at the boundary.

Therefore, A,C,D condense at the boundary.

Note : These results hold for all 2-cycle subgroups i.e., they belong to the same conjugacy class.

Finally we consider the subgroup K=$\{e, \sigma, \sigma^{-1}\}$:

```
sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[4], i, j),
               character_2_irred(G, G.conjugacy_classes()
   [0], i, j))
....:
(2, ())
(2, (1,2,3))
(2, (1,3,2))

sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[4], i, j),
               character_2_irred(G, G.conjugacy_classes()
   [1], i, j))
....:

sage: for i in G:
    for j in G:
        print (character_1_irred(G, G.subgroups()[4], i, j),
               character_2_irred(G, G.conjugacy_classes()
   [2], i, j))
....:
(2, ())
(2, ())
(2, (1,2,3))
(2, (1,3,2))
(2, (1,3,2))
(2, (1,2,3))
```

Analysis of the results is as follows : For the conjugacy class of e we have the following sum

$$2{*}tr_{\pi_i}(e) + 2{*}tr_{\pi_i}(1,2,3) + 2{*}tr_{\pi_i}(1,3,2)$$

For the representations π_1 and π_2 we have the inner product to be greater than zero, therefore V_{e,π_1}, V_{e,π_2} condense at the boundary.

None from the conjugacy class $\{\tau\}$ condense at the boundary.

While in the case of the conjugacy class $\{\sigma\}$ the inner product is as the following :

$$4{*}tr_{\pi_i}(e) + 2{*}tr_{\pi_i}(1,2,3) + 2{*}tr_{\pi_i}(1,3,2)$$

which is greater than zero only when i=1. Therefore V_{σ,π_1} condense at the boundary.

Therefore A,B,F condense at the boundary.

Therefore to summarize :
For the subgroup K=G, A, D, F condense
For the subgroup K=$\{e\}$, A, B, C condense
For the subgroup K=$\{e, \tau\}$, A, C, D condense
For the subgroup K=$\{e, \sigma, \sigma^{-1}\}$, A, B, F condense

Calculating the number of times a particular excitation condenses at the boundary

To calculate the number of times a particular excitation condenses at the boundary the following relation is used [2]:

$$\chi_{A(K)} = \Sigma_{(a,\pi)}\ \alpha_{(a,\pi)} * \chi_{(a,\pi)}$$

Consider the case of K=G, from the code table result above observe that

$$\chi_{A(G)} = \alpha_{(e,\pi_1)} * \chi_{(e,\pi_1)} + \alpha_{(\tau,\pi_1)} * \chi_{(\tau,\pi_1)} + \alpha_{(\sigma,\pi_1)} * \chi_{(\sigma,\pi_1)}$$

observing that for g=h=e, $\chi_{A(G)}(g,h) = 1$, and $\chi_{(e,\pi_1)}(g,h) = 1$, implies $\alpha_{(e,\pi_1)} = 1$ similarly $\alpha_{(\tau,\pi_1)}$, $\alpha_{(\sigma,\pi_1)}$. This is easy to see as e does not belong to the conjugacy class of $\{\tau\}$ or $\{\sigma\}$. So, A, D, F condense only once at the boundary.

Consider the case of K=$\{e\}$, from the table observe that

$$\chi_{A(G)} = \alpha_{(e,\pi_1)} * \chi_{(e,\pi_1)} + \alpha_{(e,\pi_2)} * \chi_{(e,\pi_2)} + \alpha_{(e,\pi_3)} * \chi_{(e,\pi_3)}$$

observing $\chi_{A(G)}(g,h) = 6$, and $\chi_{(e,\pi_1)}(g,h) = 1$, $\chi_{(e,\pi_2)}(g,h) = 1$, $\chi_{(e,\pi_3)}(g,h) = 2$ implies

$$\alpha_{(e,\pi_1)} + \alpha_{(e,\pi_2)} + 2^*\alpha_{(e,\pi_3)} = 6, \text{ implying } \alpha_{(e,\pi_1)} = 1,\ \alpha_{(e,\pi_2)} = 1,\ \alpha_{(e,\pi_3)} = 2$$

So, A, B condense once while C condeses twice at the boundary.

For the subgroups K=$\{e, \tau\}$, A, C, D form the condensates and

$$\alpha_{(e,\pi_1)} = 1,\ \alpha_{(e,\pi_3)} = 1,\ \alpha_{(\tau,\pi_3)} = 1$$

So A, C, D condense once at the boundary.

For the subgroups K=$\{e, \sigma, \sigma^{-1}\}$, A, B, F form the condensates and from the table

$$\alpha_{(e,\pi_1)} = 1,\ \alpha_{(e,\pi_2)} = 1,\ \alpha_{(\sigma,\pi_1)} = 2$$

So A, B condense once at the boundary, while F condenses twice at the boundary.

2.1.3 Computing the ground state with respect to different T operators on a single lattice cylinder:

Consider a single lattice i.e., identified by two vertices and a single edge. For this given construction, the ground state is computed by eigenstates of $A_v^k B_p^e T$ (equivalent to the action of $\Pi(\Sigma_v A_v^k)\Pi B_p$ on the eigenstates of opertor T). The general construction of state is given by

where g_1, $g_3 \in K$ subset of G. g_2 is a group element.

I. In the case of $T^{(e,e)} = F^{(e,e)} + F^{(e,(2,3))}$, the following form the eigenstates
For all $g_1, g_3 \in K$, g_2 is restricted to $\{e, (2,3)\}$ (from the defintion of ribbon operator in 2.1). So for each of g_2 the following condition must be satisfied (comes from the flux operator) $g_3 g_2 g_1 g_2^{-1} = e$.

So for $g_2 = e$, g_1, g_3 both are either going to $(2,3)$ or both going to e. For each of the case the eigen state (which is calculated by action of A_v at different vertices one after the other and sum of all these terms). Let the upper vertex is acted by k_u and the lower vertex by k_d both of which come from the subgroup K. So there are four combinations for a given $((k_u, k_d) = \{(e,e), (e,(2,3)), ((2,3),e), ((2,3),(2,3))\})$ state which need to be added. The action on these vertices gives a variation of g_1, g_2, g_3 which get mapped to $k_u g_1 k_u^{-1}, k_d g_2 k_u^{-1}, k_d g_3 k_d^{-1}$. To evaluate this sum the following snippet has been used :

```
sage: def ground_state(g1, g2, g3, ku, kd):
    return ku*g1*ku^-1, kd*g2*ku^-1, kd*g3*kd^-1

# for g_2 = e, g_1 = (2,3), g_3 = (2,3)
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[4], G[0], G[4], i, j)
....:
((2,3), (), (2,3))
((2,3), (2,3), (2,3))
((2,3), (2,3), (2,3))
((2,3), (), (2,3))
```

So for $g_2 = e, g_1 = g_3 = (2,3)$, the sum is the following : $\tau*e*\tau+\tau*\tau*\tau+\tau*\tau*\tau+\tau*e*\tau$

Similarly for the case $g_2 = e, g_1 = g_3 = e$,

```
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[0], G[0], G[0], i, j)
....:
((), (), ())
```

```
((), (2,3), ())
((), (2,3), ())
((), (), ())
```

So for $g_2 = e, g_1 = g_3 = e$, the sum is $e*e*e + e*\tau*e + e*\tau*e + e*e*e$

Similarly for the case $g_2 = (2,3), g_1 = g_3 = e$,

```
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[0], G[4], G[0], i, j)
....:
((), (2,3), ())
((), (), ())
((), (), ())
((), (2,3), ())
```

So for $g_2 = (2,3), g_1 = g_3 = e$, $e*\tau*e + e*e*e + e*e*e + e*\tau*e$

Similarly for the case $g_2 = (2,3), g_1 = g_3 = (2,3)$,

```
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[4], G[4], G[4], i, j)
....:
((2,3), (2,3), (2,3))
((2,3), (), (2,3))
((2,3), (), (2,3))
((2,3), (2,3), (2,3))
```

So for $g_2 = (2,3), g_1 = g_3 = e$, $\tau*\tau*\tau + \tau*e*\tau + \tau*e*\tau + \tau*\tau*\tau$

So in this case we have 2 ground states.

II. In the case of $T^{(e,(1,2))} = F^{(e,(1,2))} + F^{(e,(1,2,3))}$, the eigenstates look as follows :
For all $g_1, g_3 \in$ K, g_2 is restricted to $\{(1,2),(1,2,3)\}$ (the condition on F operator). So for each of g_2 the following condition must be satisfied (comes from the flux operator) $g_3 g_2 g_1 g_2^{-1} = e$.

So for $g_2 = (1,2)$, $g_1 = g_3 = e$,

```
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[0], G[1], G[0], i, j)
....:
((), (1,2), ())
((), (1,2,3), ())
((), (1,3,2), ())
((), (1,3), ())
```

So for $g_2 = (1,2), g_1 = g_3 = e$, $e*(1,2)*e + e*(1,2,3)*e + e*(1,3,2)*e + e*(1,3)*e$

Similarly for $g_2 = (1,2,3)$, $g_1 = g_3 = e$,

```
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[0], G[2], G[0], i, j)
....:
((), (1,2,3), ())
((), (1,2), ())
((), (1,3), ())
((), (1,3,2), ())
```

Hence the sum is $e*(1,2)*e + e*(1,2,3)*e + e*(1,3,2)*e + e*(1,3)*e$

III. In the case of $T^{(e,(1,2,3))} = F^{(e,(1,3))} + F^{(e,(1,3,2))}$, following are the eigenstates
For all $g_1, g_3 \in$ K, g_2 is restricted to $\{(1,3),(1,3,2)\}$. So for each of g_2 the following condition must be satisfied (comes from the flux operator) $g_3 g_2 g_1 g_2^{-1} = e$.

So for $g_2 = (1,3)$, $g_1 = g_3 = e$,

```
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[0], G[5], G[0], i, j)
....:
((), (1,3), ())
((), (1,3,2), ())
```

```
((), (1,2,3), ())
((), (1,2), ())
```

So for $g_2 = (1,3), g_1 = g_3 = e$, $e*(1,2)*e + e*(1,2,3)*e + e*(1,3,2)*e + e*(1,3)*e$

Similarly for $g_2 = (1,3,2)$, $g_1 = g_3 = e$,

```
sage: for i in G.subgroups()[1]:
    for j in G.subgroups()[1]:
        print ground_state(G[0], G[3], G[0], i, j)
....:
((), (1,3,2), ())
((), (1,3), ())
((), (1,2), ())
((), (1,2,3), ())
```

Hence the sum is $e*(1,2)*e + e*(1,2,3)*e + e*(1,3,2)*e + e*(1,3)*e$

Similarly for the other three operators, the same linear combination of states is observed, thus for the boundary case $K = \{e, \tau\}$ we have 3 unique ground states.

For the calculation of the number of excitations, condensates on the boundary, ribbon operators, and ground states in a more general setting refer to Appendix A.

Chapter 3

Category Theory and Lattice Models

3.1 Levin-Wen Models

The aim of this section is to introduce the most general structure used to classify topological phases of matter. This uses the concepts presented in Chapter 1 and it will be shown that the contents presented in Chapter 2 will form a subclass of these models in Chapter 4. The String Net Model or Levin-Wen Model [6] is defined on trivalent graph embedded to a closed oriented surface. The rules of the model are as follows :

1 String types : They are finite number of string types, and the set of string types is isomorphic to positive integers.

2 Branching rules : Only certain string types are allowed to form a vertex.

3 String orientation : Every string has a dual type, which is directed in the opposite direction.

The universal features of the string net model are given by fig 3.1 :

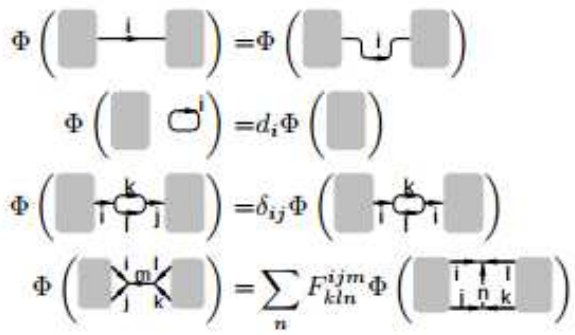


FIGURE 3.1: Universal features of the String-Net Model

Equivalently, they are captured by six index object F_{ijk}^{klm} and the numbers d_i, that is the F moves and the quantum dimensions. However not all combinations of these give rise to string net model as they are constrainted by the above equations. The only valid combinations are those which satisfy :

$$F_{j^*i^*0}^{ijk} = \frac{\sqrt{d_k}}{\sqrt{d_i d_k}}\delta_{ijk}$$
$$F_{kln}^{ijm} = F_{jin}^{klm^*} = F_{lkn^*}^{jim} = F_{k^*ln}^{imj}\frac{\sqrt{d_m d_n}}{\sqrt{d_j d_l}}$$
$$\Sigma_{n=0}^{N} F_{kp^*n}^{mlq} F_{mns^*}^{jip} F_{lkr^*}^{js^*n} = F_{q^*kr^*}^{jip} F_{mls^*}^{riq^*}$$

Unitary Tensor Categories are the fundamental framework for the string net model. The string labels form the objects in category, the Homspace can be seen as branching rules. Given a group G, the string labels are the irreducible representations of the group G, the quantum dimension d_i is the dimension of the representation, and the F object is the $6j$ symbols of the group.

For every valid (F_{lmn}^{ijk}, d_i), the Hamiltonian of the model is given by :

$$H = -\sum_I Q_I - \sum_p B_p, \text{ where } B_p = \sum_{s=0}^{N} a_s B_p^s$$

where Q_I measures the electric charge and favors no charge configuration, and B_p measures the magnetic flux through a plaquette and favors no flux. The figure 3.2 give the action of Q_I and $B_{\mathbf{p}}$ on a arbitrary lattice. Q_I and B_p^s commute with each other, making the Hamiltonian exactly soluble. The ground state satisfies $Q_I = B_p^s = 1$ for all I, p while the excited states are those which violate these conditions.

Given the strings are labelled by UTC C, the excitations are given by the monoidal center $Z(C)$ of the UTC C, which is a Modular Tensor Category. For example, consider

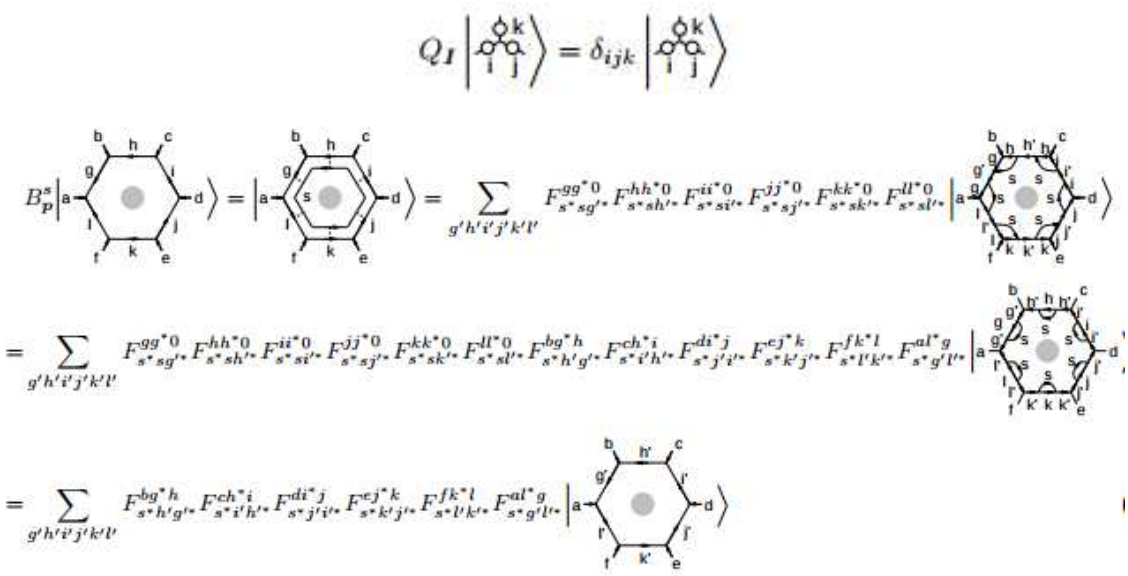


FIGURE 3.2: Definition of the Q_I and B_p operators

the Toric code Z_2, the strings are indexed by one dimensional representations, say, 1 and -1 and the monoidal center of is given by the pair (M, ρ) where ρ is given by $\rho : \_ \otimes M \to M \otimes \_$, which in this case results in a category with rank four, these four form the excitations in the Toric Code.

3.2 Boundary construction, Condensations in String-Net model

The following work by Kitaev and Kong [7], presents the relationship between the UTC C and the boundary labels. Consider a lattice with boundary, with edges being labelled by M, which should satisfy all conditions mentioned in the previous section for a valid string-net configuration, that is they should be compatible with the F moves. Such a structure of the boundary is provided by the left module category over the UTC C. Therefore the edge labels on the boundary M are given by the objects of left module category over the UTC C. Given the bulk labels and the boundary labels it is possible to provide the Hamiltonian of the lattice. Once the Hamiltonian is defined, it should be possible to compute the ground state, which is used to compute the topological entanglement entropy.

Given the bulk of the lattice is labelled by simple objects from UTC C, the excitations are given by the center $Z(C)$ of the UTC C, which is a Modular Tensor Category. Using the above construction of Modular Tensor Category, the construction of the excitations on the boundary, the construction of the condensed phase category can be achieved, as mentioned below. For the detailed proof, refer to [4]

Consider the following construction, the excitations of a particular lattice given by UMTC C, the boundary excitations by UTC E, the condensed phase by another UMTC D.

For one step condesations the following results hold :

1 Vacuum of D is given by condensable algebra A in C. (condensable implies the algebra is connected, commutative and separable)

2 $D \simeq C_A^{loc}$ category of local right A-modules in C.

3 $E \simeq C_A$ category of right A-modules in C.

4 Anyons in the bulk move onto the wall by the following functor map :

$$\_ \otimes A : C \to C_A$$

For two step condensations the following results hold :

1 Vacuum in D is given by condensable algebra A in C.

2 D is given by local right A-modules

3 Vacuum in E is given by connected separable algebra B in C.

4 E is given by B-bimodule

5 Bulk to wall map from the C side is given by :

$$\_ \otimes B : C \to : C_{B|B}$$

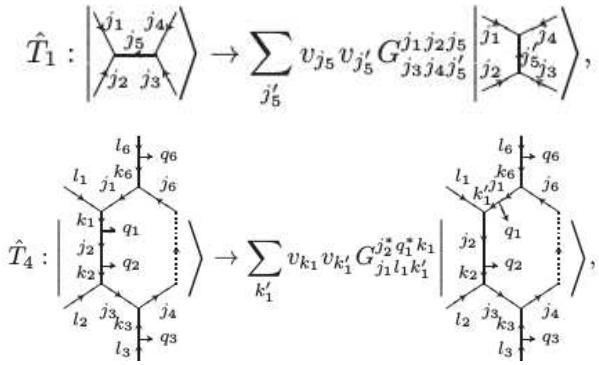


FIGURE 3.3: Invariant operators in generalized Levin-Wen Model

The above statements provide an abstract insight into the relationship between the excitations in the bulk, on the boundary and the condensed phase. Though given the Modular Tensor Category, it is not totally possible to construct a lattice model with boundary, as there is no information of the labels in the bulk (the same MTC can be viewed as a center of different UTC which are equivalent upto Morita equivalence) and also the labels on the boundary.

3.3 Excitations, String operators in String-Net Model

The excitations in the model are those eigenvectors which do not satisfy the ground state conditions, if $Q_v = 0$ is violated the excitation is a charge excitation at vertex v and is identified by a non-trivial edge label attached to the vertex and if $B_p^s = 0$ is violated a fluxon excitation is identified at the plaquette p. If all the tail labels are trivial, Levin-Wen lattice is recovered. The lattice with charges are invariant under the operators as defined in figure 3.3 : The first one is the F move and the second is used to move the non-trivial label around the particular vertex.

The string operator $W_e^{J;pq^*}$ acting on a state, giving rise to the J fluxon and charges p and q^*, is as given in figure 3.4

String operator is the ribbon operator equivalent in the String-Net model. For various other operations on the string operators, the charge measurement, the flux measurement and the action of Q_v^q and B_p^s refer to [8].

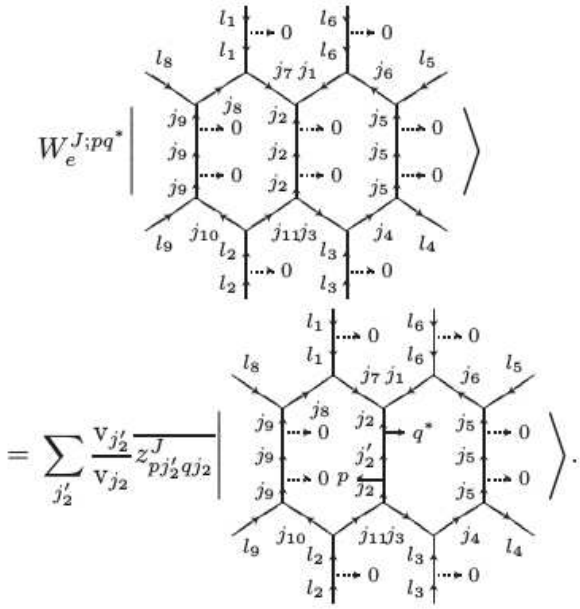


FIGURE 3.4: String operator in generalized Levin-Wen Model

Chapter 4

Quantum Doubles and other examples in String-Net picture

4.1 Toric Code, Quantum Double of S_3 in terms of Categories

As mentioned in the previous Chapter 3, the edge labelling is done by irreducible representations of the group, which forms a Unitary Tensor Category. The excitations are given by the monoidal center of the Unitary Tensor Category which is a Modular Tensor Category. This section aims to present some examples to realize the above statements in case of Toric Code, $D(Z_2)$ and $D(S_3)$.

$D(Z_2)$, Toric Code:
The input to the Toric Code is the group Z_2, that is the edges are indexed by $-1, 1$. Z_2 has two one dimensional irreducible representations. These form the data for the Unitary Tensor Category. The excitations are given by the center of the unitary tensor category. The objects in the center of the category C are given by a pair (M, ρ_X) where $\rho_X : X \otimes M \to M \otimes X$, where $M, X \in C$. The irreducible representations of Z_2, say V_1 and V_{-1} each branching into (V_1, ρ_1), (V_1, ρ_{-1}), (V_{-1}, ρ_1), (V_{-1}, ρ_{-1}). Therefore the rank of the modular tensor category is four and thus there are four excitations.

$D(S_3)$:
The input to Quantum Double of S_3 is the group S_3. S_3 has two one dimensional and

one two dimensional irreducible representations. These form the data for the Unitary Tensor Catgory. The excitations are given by the center of the representation category of S_3, which is isomorphic to the irreducible representations of the Drinfeld Double, which is indexed by irreducible representations of the centralizer of the conjugacy classes of the group. In this example, there are eight such objects. Each of the irreducible representation is a simple object in the Modular Tensor Category.

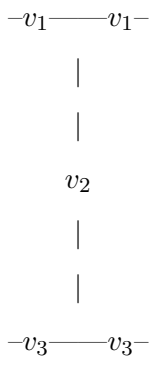
4.2 Boundary construction for Toric Code, $D(S_3)$

The boundary labels are given by objects of the left modules over C, C_M, where C is a Unitary Tensor Category used to index the bulk. Consider the case of Kitaev Quantum Double $D(S_3)$, the boundary labels were given by the subgroup $K \subset S_3$, from the definition of left module over a Unitary Tensor Category, it is easy to that the subgroups form the left modules. Thereby, the boundaries are indexed by various elements of the subgroups.

Consider the following construction, the excitations given by MTC C, the boundary excitations by E, the condensed phase by D. The boundary excitations are given by right A-modules in C, where C is the Modular Tensor Category whose objects are the excitations of the lattice model. The vacuum in D is a condensable algebra A in C. For the toric code, suppose the excitations are labelled by $1, e, m, f$. $1 \oplus e$ and $1 \oplus m$ both are objects in C and are algebras in C. This is easy to see from the graphical calculus viewpoint and also verifying the commutation relationships. These in addition of being algebras, also satisfy the connected, commutative, separable properties making them condensable algebras. Hence these form the vacuum in the condensed phase. The detection of these algebras sets the construction of not only E and D, but also the excitations which condense on the boundary. Referring to the functor, the excitation to the wall map, in the case of Toric Code for the condensable algebra $1 \oplus e$, the excitations $1, e$ are identified with $1 \oplus e$ and f, m are identified with $m \oplus e$. D itself is given by $1 + e$, and E is given by $m \oplus e$ and $1 \oplus e$. The above construction provides an insight into the excitations on the boundary, as well the excitations which condense on the boundary, but not the boundary labels. The above techniques can be extended to obtain the bulk excitations, the boundary excitations and the condensed phase for $D(S_3)$ using the data from [9].

4.3 Ground state construction for lattice with Ising on boundary

Consider the following lattice model, the edges coming from the UTC of Ising whose objects are $1, \psi, \sigma$. The boundary labels also come from the UTC of Ising, as it forms a left-module of the bulk, which is Ising. The ground state is both the eigen-states of the operators Q_v and B_p, where Q_v are fusion rules and B_p is given by $\Sigma(d_k/D^2)B_p^k$. Consider a single lattice on the cylinder.



$(v1, v2, v3) = \{\{1,1,1\} - (1), \{\psi,1,\psi\} - (2), \{\psi,1,1\} - (3), \{\sigma,1,\psi\} - (4), \{\sigma,1,\sigma\} - (5), \{\sigma,1,1\} - (6), \{\sigma,\psi,\sigma\} - (7), \{\psi,1,\sigma\} - (8), \{1,1,\sigma\} - (9), \{1,1,\psi\} - (10)\}$

The action of B_p^1 on the above set results in the same set.
The action of B_p^ψ on the above set results in the following $\{2,1,10,6,5,4,7,9,8,3\}$.
The action of B_p^σ on the above set results in the following 1/sqrt(2) times $\{5+7, 5+7, 5-7, 8+9, 1+2+3+10, 8+9, 1+2-3-10, 4+6, 4+6, 3-7\}$.

The matrices $B_p^1, B_p^\psi, B_p^\sigma$ are to used compute B_p which is given by $\frac{1}{D}\sum_s d_s B_p^s$, which is further used to compute the ground states:

```
julia> B_p_1 = eye(10);


julia> B_p_psi = [[0 1 0 0 0 0 0 0 0 0]; [1 0 0 0 0 0 0 0 0
   0]; [0 0 0 0 0 0 0 0 0 1];
```

```
                  [0 0 0 0 0 1 0 0 0 0]; [0 0 0 0 1 0 0 0 0
   0]; [0 0 0 1 0 0 0 0 0 0];
                  [0 0 0 0 0 0 1 0 0 0]; [0 0 0 0 0 0 0 0 1
   0]; [0 0 0 0 0 0 0 1 0 0];
                  [0 0 1 0 0 0 0 0 0 0]];

julia> B_p_s = [[0 0 0 0 1 0 1 0 0 0]; [0 0 0 0 1 0 1 0 0
   0]; [0 0 0 0 1 0 -1 0 0 0];
                [0 0 0 0 0 0 0 1 1 0]; [1 1 1 0 0 0 0 0 0
   1]; [0 0 0 0 0 0 0 1 1 0];
                [1 1 -1 0 0 0 0 0 0 -1]; [0 0 0 1 0 1 0 0 0
   0]; [0 0 0 1 0 1 0 0 0 0];
                [0 0 0 0 1 0 -1 0 0 0]]

julia> B_p_sigma = 1/sqrt(2)*B_p_s;

julia> B = (1/4)*(B_p_1 + B_p_psi + sqrt(2)*B_p_sigma);

julia> eigvals(inv(eigvecs(B))*B*eigvecs(B))
10-element Array{Complex{Float64},1}:
         -7.91813e-18+0.0im
                  1.0+0.0im
8.82083e-18+8.37086e-18im
8.82083e-18-8.37086e-18im
                 1.0+0.0im
                 1.0+0.0im
        -7.14582e-18+0.0im
1.27443e-18+1.57887e-18im
1.27443e-18-1.57887e-18im
         1.42363e-18+0.0im

julia> eigvecs(inv(eigvecs(B))*B*eigvecs(B))
10x10 Array{Complex{Float64},2}:
```

```
%
```

Thus, the number of ground states of the lattice with Ising both on bulk and boundary is three, which is in agreement with [10].

Part II

Chapter 5

Gapped Boundaries, Fold and Unfold Relations

5.1 Map between the excitations of the Quantum Double $G \times G^{'}$ - The unfolded and folded versions

The excitations of a Quantum Double indexed by $G \times G^{'}$ are given by the irreducible representations of the centralizer of the conjugacy classes of $G \times G^{'}$. The inner product between the character associated with the excitation and the character associated with the boundary indexed by the subgroup is used to identify the condensate on the boundary leading to the classification of the boundaries itself. The character, $\chi_{G\times G'}$ associated with the excitation now depends on the elements of $G \times G^{'}$ and is equivalent to $\chi_G \times \chi_{G'}$. Therefore, leading to the identification of condensate on the boundary and hence in the unfolded version the excitations can be mapped on either side of the boundary turned domain wall of $G \times G^{'}$.

5.2 Folded $Z_2 \times Z_2$, unfolding and mapping excitations in the unfolded domain

The different boundary conditions of $Z_2 \times Z_2$ are identified by the subgroup and the 2-cocycle. Using the condition of inner product of characters from the paper by Beigi

et al [2], we identify the condensates on various boundaries from the following data :

```
 def character_excitation(G, conjugacy_class, g, h):
    k_h = 0
    for g_1 in gap.AsList(G):
        if h*g_1 == g_1*conjugacy_class.Representative():
            k_h = g_1
            break
    if g*h == h*g and k_h != 0:
        return gap.Inverse(k_h)*g*k_h
    else:
        return 0

def character_subgroup(G, subgroup, g, h):
    cocycle = []
    if h*g == g*h:
        for g_1 in gap.AsList(G):
              if g_1*g*gap.Inverse(g_1) in gap.AsList(
   subgroup) and
               g_1*h*gap.Inverse(g_1) in gap.AsList(subgroup
   ):
                # the below condition is to be used for a
   general case,
                # cocycle.append([(g_1*g*gap.Inverse(g_1),
   g_1*h*gap.Inverse(g_1)),(g_1*g*h*gap.Inverse(g_1), g_1*
   gap.Inverse(g)*gap.Inverse(g_1))])
                # however for abelian cases the cocycle
   reduces to the following,
                cocycle.append([(g, h),(h, g)])
    return cocycle

def inner_product_of_characters(QDM_group, subgroup,
   conjugacy_class):
    inner_product_terms = []
```

```
    for g in gap.AsList(QDM_group):
        for h in gap.AsList(QDM_group):
            char_sub = character_subgroup(QDM_group,
   subgroup, g, h)
            if len(char_sub) != 0 and character_excitation(
   QDM_group, conjugacy_class, g, h) != 0:
                inner_product_terms.append(["g : ", g, "h :"
   , h, "character_subgroup :",char_sub, "length_subgroup :"
    ,len(gap.AsList(subgroup)),"character_excitation :",
   character_excitation(QDM_group, conjugacy_class, g, h)])
    return inner_product_terms

def summarize(filename, group):
    f = open(filename,'w')
    for subgroup in range(1,len(gap.AsList(gap.AllSubgroups(
   group)))+1):
        print >> f, "subgroup : ",gap.AsList(gap.AsList(gap.
   AllSubgroups(group))[subgroup])
        for conj in range(1,len(gap.AsList(group.
   ConjugacyClasses()))+1):
            print >> f, "conjugacy_class :", gap.AsList(gap.
   AsList(group.ConjugacyClasses())[conj]), "innerproduct :"
   ,inner_product_of_characters(group, gap.AsList(gap.
   AllSubgroups(group))[subgroup], gap.AsList(group.
   ConjugacyClasses())[conj])
        print  >> f,"\n"
```

The output of the above code is presented below

```
subgroup :  [ () ]

conjugacy_class : [ () ]
innerproduct :
[['g : ', (), 'h :', (),
```

```
'character_subgroup :',
[[(), ()), (), ())], [(), ()), (), ())],
[(), ()), (), ())], [(), ()), (), ())]],
'length_subgroup :', 1, 'character_excitation :', ()]]

conjugacy_class : [ (3,4) ] innerproduct : []

conjugacy_class : [ (1,2) ] innerproduct : []

conjugacy_class : [ (1,2)(3,4) ] innerproduct : []
```

The inner product here can be read as $\frac{4}{1}tr_{\pi_i}(e)$, where π_i is the Irrep of the centralizer of the conjugacy class of (), where the centralizer of the conjugacy class of () is equivalent to the group $Z_2 \times Z_2$ itself. Using the statements on character from the previous section, the inner product is equivalent to $4tr_{\pi_i}(e)_{Z_2} tr_{\pi_i}(e)_{Z_2}$.

The trace of the irreps of the conjugacy class of e is given by the character table

Irrep	()	$(1,2)$
π_1	1	1
π_2	1	-1

the rows give the trace of the irreps, while the conjugacy classes are identified by the columns.

Therefore, the condensate is given by $1 \oplus e_1 \oplus e_2 \oplus e_1 e_2$

```
subgroup :  [ (), (3,4) ]

conjugacy_class : [ () ]
innerproduct :
[['g : ', (), 'h :', (),
'character_subgroup :',
[[(), ()), (), ())], [(), ()), (), ())],
[(), ()), (), ())], [(), ()), (), ())]],
'length_subgroup :', 2, 'character_excitation :', ()],
['g : ', (3,4), 'h :', (),
```

```
'character_subgroup :',
[[(3,4), ()), ((), (3,4))], [(3,4), ()), ((), (3,4))],
[(3,4), ()), ((), (3,4))], [(3,4), ()), ((), (3,4))]],
'length_subgroup :', 2, 'character_excitation :', (3,4)]]

conjugacy_class : [ (3,4) ]
innerproduct :
[['g : ', (), 'h :', (3,4),
'character_subgroup :',
[[(), (3,4)), ((3,4), ())], [(), (3,4)), ((3,4), ())],
[(), (3,4)), ((3,4), ())], [(), (3,4)), ((3,4), ())]],
'length_subgroup :', 2, 'character_excitation :', ()],
['g : ', (3,4), 'h :', (3,4),
'character_subgroup :',
[[(3,4), (3,4)), ((3,4), (3,4))], [(3,4), (3,4)), ((3,4),
   (3,4))],
[(3,4), (3,4)), ((3,4), (3,4))], [(3,4), (3,4)), ((3,4),
   (3,4))]],
'length_subgroup :', 2, 'character_excitation :', (3,4)]]

conjugacy_class : [ (1,2) ] innerproduct : []

conjugacy_class : [ (1,2)(3,4) ] innerproduct : []
```

The inner product here can be read as $\frac{4}{2}tr_{\pi_i}() + \frac{4}{2}tr_{\pi_i}(3,4)$, which is equivalent to $2tr_{\pi_i}(e)_{Z_2}tr_{\pi_i}(e)_{Z_2} + 2tr_{\pi_i}(e)_{Z_2}tr_{\pi_i}(3,4)_{Z_2}$

Irrep	()	$(1,2)$
π_1	1	1
π_2	1	-1

Irrep	()	$(3,4)$
π_3	1	1
π_4	1	-1

Therefore, the condensates are $\{1, e_1, m_2, e_1 m_2\}$.

```
subgroup :  [ (), (1,2) ]

conjugacy_class : [ () ]
innerproduct :
[['g : ', (), 'h :', (),
'character_subgroup :',
[[(), ()), (), ())], [(), ()), (), ())],
[(), ()), (), ())], [(), ()), (), ())]],
'length_subgroup :', 2, 'character_excitation :', ()],
['g : ', (1,2), 'h :', (),
'character_subgroup :',
[[(1,2), ()), (), (1,2))], [(1,2), ()), (), (1,2))],
[(1,2), ()), (), (1,2))], [(1,2), ()), (), (1,2))]],
'length_subgroup :', 2, 'character_excitation :', (1,2)]]

conjugacy_class : [ (3,4) ] innerproduct : []

conjugacy_class : [ (1,2) ]
innerproduct :
[['g : ', (), 'h :', (1,2),
'character_subgroup :',
[[(), (1,2)), ((1,2), ())], [(), (1,2)), ((1,2), ())],
[(), (1,2)), ((1,2), ())], [(), (1,2)), ((1,2), ())]],
'length_subgroup :', 2, 'character_excitation :', ()],
['g : ', (1,2), 'h :', (1,2),
'character_subgroup :',
[[(1,2), (1,2)), ((1,2), (1,2))], [(1,2), (1,2)), ((1,2),
   (1,2))],
[(1,2), (1,2)), ((1,2), (1,2))], [(1,2), (1,2)), ((1,2),
   (1,2))]],
'length_subgroup :', 2, 'character_excitation :', (1,2)]]
```

```
conjugacy_class : [ (1,2)(3,4) ] innerproduct : []
```

The innerproduct here can be read as $\frac{4}{2}tr_{\pi_i}()+\frac{4}{2}tr_{\pi_i}(1,2)$ which is equivalent to $2tr_{\pi_i}(e)_{Z_2}tr_{\pi_i}(e)_{Z_2}$ + $2tr_{\pi_i}(1,2)_{Z_2}tr_{\pi_i}(e)_{Z_2}$

Therefore, the condensates are $\{1, e_2, m_1, e_2m_1\}$

```
subgroup :  [ (), (1,2)(3,4) ]

conjugacy_class : [ () ]
innerproduct :
[['g : ', (), 'h :', (),
'character_subgroup :',
[[(), ()), (), ())], [(), ()), (), ())],
[(), ()), (), ())], [(), ()), (), ())]],
'length_subgroup :', 2, 'character_excitation :', ()],
['g : ', (1,2)(3,4), 'h :', (),
'character_subgroup :',
[[((1,2)(3,4), ()), (), (1,2)(3,4))], [((1,2)(3,4), ()),
   (), (1,2)(3,4))],
[((1,2)(3,4), ()), (), (1,2)(3,4))], [((1,2)(3,4), ()), (()
   , (1,2)(3,4))]],
'length_subgroup :', 2, 'character_excitation :', (1,2)(3,4)
   ]]

conjugacy_class : [ (3,4) ] innerproduct : []

conjugacy_class : [ (1,2) ] innerproduct : []

conjugacy_class : [ (1,2)(3,4) ]
innerproduct :
[['g : ', (), 'h :', (1,2)(3,4),
'character_subgroup :',
[[(), (1,2)(3,4)), ((1,2)(3,4), ())], [(), (1,2)(3,4)),
   ((1,2)(3,4), ())],
```

```
[((), (1,2)(3,4)), ((1,2)(3,4), ())], [((), (1,2)(3,4)),
   ((1,2)(3,4), ())]],
'length_subgroup :', 2, 'character_excitation :', ()],
['g : ', (1,2)(3,4), 'h :', (1,2)(3,4),
'character_subgroup :',
[[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))],
[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))],
[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))],
[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))]],
'length_subgroup :', 2, 'character_excitation :', (1,2)(3,4)
   ]]
```

The innerproduct here can be read as $\frac{4}{2}tr_{\pi_i}() + \frac{4}{2}tr_{\pi_i}((1,2)(3,4))$ which is equivalent to $2tr_{\pi_i}(e)_{Z_2}tr_{\pi_i}(e)_{Z_2} + 2tr_{\pi_i}(1,2)_{Z_2}tr_{\pi_i}(3,4)_{Z_2}$

Therefore, the condensates here are $\{1, e_1e_2, m_1m_2, f_1f_2\}$

```
subgroup :  [ (), (3,4), (1,2), (1,2)(3,4) ]


conjugacy_class : [ () ]
innerproduct :
[['g : ', (), 'h :', (),
'character_subgroup :',
[[(), ()), (), ())], [(), ()), (), ())],
[(), ()), (), ())], [(), ()), (), ())]],
'length_subgroup :', 4, 'character_excitation :', ()],
['g : ', (3,4), 'h :', (),
'character_subgroup :',
[[(3,4), ()), (), (3,4))], [(3,4), ()), (), (3,4))],
[(3,4), ()), (), (3,4))], [(3,4), ()), (), (3,4))]],
'length_subgroup :', 4, 'character_excitation :', (3,4)],
['g : ', (1,2), 'h :', (),
'character_subgroup :',
[[(1,2), ()), (), (1,2))], [(1,2), ()), (), (1,2))],
[(1,2), ()), (), (1,2))], [(1,2), ()), (), (1,2))]],
```

```
'length_subgroup :', 4, 'character_excitation :', (1,2)],
['g : ', (1,2)(3,4), 'h :', (),
'character_subgroup :',
[[((1,2)(3,4), ()), (), (1,2)(3,4))], [((1,2)(3,4), ()),
   (), (1,2)(3,4))],
[((1,2)(3,4), ()), (), (1,2)(3,4))], [((1,2)(3,4), ()), ()
   , (1,2)(3,4))]],
'length_subgroup :', 4, 'character_excitation :', (1,2)(3,4)
   ]]

conjugacy_class : [ (3,4) ]
innerproduct :
[['g : ', (), 'h :', (3,4),
'character_subgroup :',
[[(), (3,4)), ((3,4), ())], [(), (3,4)), ((3,4), ())],
[(), (3,4)), ((3,4), ())], [(), (3,4)), ((3,4), ())]],
'length_subgroup :', 4, 'character_excitation :', ()],
['g : ', (3,4), 'h :', (3,4),
'character_subgroup :',
[[(3,4), (3,4)), ((3,4), (3,4))], [(3,4), (3,4)), ((3,4),
   (3,4))],
[(3,4), (3,4)), ((3,4), (3,4))], [(3,4), (3,4)), ((3,4),
   (3,4))]],
'length_subgroup :', 4, 'character_excitation :', (3,4)],
['g : ', (1,2), 'h :', (3,4),
'character_subgroup :',
[[(1,2), (3,4)), ((3,4), (1,2))], [(1,2), (3,4)), ((3,4),
   (1,2))],
[(1,2), (3,4)), ((3,4), (1,2))], [(1,2), (3,4)), ((3,4),
   (1,2))]],
'length_subgroup :', 4, 'character_excitation :', (1,2)],
['g : ', (1,2)(3,4), 'h :', (3,4),
'character_subgroup :',
```

```
[[((1,2)(3,4), (3,4)), ((3,4), (1,2)(3,4))], [((1,2)(3,4),
   (3,4)), ((3,4), (1,2)(3,4))],
[((1,2)(3,4), (3,4)), ((3,4), (1,2)(3,4))], [((1,2)(3,4),
   (3,4)), ((3,4), (1,2)(3,4))]],
'length_subgroup :', 4, 'character_excitation :', (1,2)(3,4)
   ]]

conjugacy_class : [ (1,2) ]
innerproduct :
[['g : ', (), 'h :', (1,2),
'character_subgroup :',
[[(), (1,2)), ((1,2), ())], [(), (1,2)), ((1,2), ())],
[(), (1,2)), ((1,2), ())], [(), (1,2)), ((1,2), ())]],
'length_subgroup :', 4, 'character_excitation :', ()],
['g : ', (3,4), 'h :', (1,2),
'character_subgroup :',
[[(3,4), (1,2)), ((1,2), (3,4))], [(3,4), (1,2)), ((1,2),
   (3,4))],
[(3,4), (1,2)), ((1,2), (3,4))], [(3,4), (1,2)), ((1,2),
   (3,4))]],
'length_subgroup :', 4, 'character_excitation :', (3,4)],
['g : ', (1,2), 'h :', (1,2), 'character_subgroup :',
[[(1,2), (1,2)), ((1,2), (1,2))], [(1,2), (1,2)), ((1,2),
   (1,2))],
[(1,2), (1,2)), ((1,2), (1,2))], [(1,2), (1,2)), ((1,2),
   (1,2))]],
'length_subgroup :', 4, 'character_excitation :', (1,2)],
['g : ', (1,2)(3,4), 'h :', (1,2), 'character_subgroup :',
[[(1,2)(3,4), (1,2)), ((1,2), (1,2)(3,4))], [(1,2)(3,4),
   (1,2)), ((1,2), (1,2)(3,4))],
[(1,2)(3,4), (1,2)), ((1,2), (1,2)(3,4))], [(1,2)(3,4),
   (1,2)), ((1,2), (1,2)(3,4))]],
```

```
'length_subgroup :', 4, 'character_excitation :', (1,2)(3,4)
   ]]



conjugacy_class : [ (1,2)(3,4) ]
innerproduct :
[['g : ', (), 'h :', (1,2)(3,4),
'character_subgroup :',
[[(), (1,2)(3,4)), ((1,2)(3,4), ())], [(), (1,2)(3,4)),
   ((1,2)(3,4), ())],
[(), (1,2)(3,4)), ((1,2)(3,4), ())], [(), (1,2)(3,4)),
   ((1,2)(3,4), ())]],
'length_subgroup :', 4, 'character_excitation :', ()],
['g : ', (3,4), 'h :', (1,2)(3,4),
'character_subgroup :',
[[(3,4), (1,2)(3,4)), ((1,2)(3,4), (3,4))], [(3,4), (1,2)
   (3,4)), ((1,2)(3,4), (3,4))],
[(3,4), (1,2)(3,4)), ((1,2)(3,4), (3,4))], [(3,4), (1,2)
   (3,4)), ((1,2)(3,4), (3,4))]],
'length_subgroup :', 4, 'character_excitation :', (3,4)],
['g : ', (1,2), 'h :', (1,2)(3,4),
'character_subgroup :',
[[(1,2), (1,2)(3,4)), ((1,2)(3,4), (1,2))], [(1,2), (1,2)
   (3,4)), ((1,2)(3,4), (1,2))],
[(1,2), (1,2)(3,4)), ((1,2)(3,4), (1,2))], [(1,2), (1,2)
   (3,4)), ((1,2)(3,4), (1,2))]],
'length_subgroup :', 4, 'character_excitation :', (1,2)],
['g : ', (1,2)(3,4), 'h :', (1,2)(3,4),
'character_subgroup :',
[[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))],
[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))],
[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))],
[(1,2)(3,4), (1,2)(3,4)), ((1,2)(3,4), (1,2)(3,4))]],
```

```
'length_subgroup :', 4, 'character_excitation :', (1,2)(3,4)
   ]]
```

The innerproduct for the conjugacy class of () can be read as

$$\frac{4}{4}tr_{\pi_i}() + \frac{4}{4}tr_{\pi_i}(3,4) + \frac{4}{4}tr_{\pi_i}(1,2) + \frac{4}{4}tr_{\pi_i}((1,2)(3,4) \tag{5.1}$$

which reduces to :

$$tr_{\pi_i}() + tr_{\pi_i}(3,4) + tr_{\pi_i}(1,2) + tr_{\pi_i}((1,2)(3,4) \tag{5.2}$$

The innerproduct for the conjugacy class of $(3,4)$ can be read as

$$\begin{aligned}\frac{4}{4}tr_{\pi_i}() + \frac{4}{4}tr_{\pi_i}(3,4) + \frac{4}{4}\frac{\phi((1,2),(3,4))}{\phi((3,4),(1,2)}tr_{\pi_i}(1,2)+ \\ \frac{4}{4}\frac{\phi((1,2)(3,4),(3,4))}{\phi((3,4),(1,2)(3,4))}tr_{\pi_i}((1,2)(3,4)\end{aligned} \tag{5.3}$$

which reduces to :

$$tr_{\pi_i}() + tr_{\pi_i}(3,4) - tr_{\pi_i}(1,2) - tr_{\pi_i}((1,2)(3,4) \tag{5.4}$$

The innerproduct for the conjugacy class of $(1,2)$ can be read as

$$\begin{aligned}\frac{4}{4}tr_{\pi_i}() + \frac{4}{4}\frac{\phi((3,4),(1,2))}{\phi((1,2),(3,4))}tr_{\pi_i}(3,4) + \frac{4}{4}tr_{\pi_i}(1,2)+ \\ \frac{4}{4}\frac{\phi((1,2)(3,4),(1,2))}{\phi((1,2),(1,2)(3,4))}tr_{\pi_i}((1,2)(3,4)\end{aligned} \tag{5.5}$$

which reduces to :

$$tr_{\pi_i}() - tr_{\pi_i}(3,4) + tr_{\pi_i}(1,2) - tr_{\pi_i}((1,2)(3,4) \tag{5.6}$$

The innerproduct for the conjugacy class of $(1,2)(3,4)$ can be read as

$$\begin{aligned}\frac{4}{4}tr_{\pi_i}() + \frac{4}{4}\frac{\phi((3,4),(1,2)(3,4))}{\phi((1,2)(3,4),(3,4))}tr_{\pi_i}(3,4)+ \\ \frac{4}{4}\frac{\phi((1,2),(1,2)(3,4))}{\phi((1,2)(3,4),(1,2))}tr_{\pi_i}(1,2) + \frac{4}{4}tr_{\pi_i}((1,2)(3,4)\end{aligned} \tag{5.7}$$

which reduces to :

$$tr_{\pi_i}() - tr_{\pi_i}(3,4) - tr_{\pi_i}(1,2) + tr_{\pi_i}((1,2)(3,4) \tag{5.8}$$

The group cohomology $H^2(Z_2^2, U(1)) = Z_2$ has 2-cocycles [11][12], given by

$$\phi^{n_1}(b,c) = exp(i\pi n_1(b_1 c_2)),$$

where (b,c) is the ordered pair given by $((b_1,b_2),(c_1,c_2))$ and $b_1,b_2,c_1,c_2,n_1 \in \{0,1\}$

For $n_1 = 0$, $\phi^0(b,c) = 1 \forall ((b_1,c_1),(b_2,c_2)) \in Z_2^2$. Applying ϕ^0 to (5.3), (5.5), (5.7) reduces to (5.2), implying for the trivial cocycle $1, m_1, m_2, m_1 m_2$ condense on the boundary.

The value of the non-trivial co-cycle is as follows for $(Z_2, +)$ and $(Z_2, \times)$:

For $\{((0,\_),(\_,0)),((1,\_),(\_,0)),((0,\_),(\_,1))\}$ the cocycle evalutes to 1 and for $\{((1,\_),(\_,1))\}$ evalutes to -1.
Equivalently $\{((1,\_),(\_,1)),((-1,\_),(\_,1)),((1,\_),(\_,-1))\}$ the cocycle evalutes to 1 and for $\{((-1,\_),(\_,-1))\}$ evalutes to -1.

For the non-trivial cocycle $\{1, e_1 m_2, m_1 e_2, f_1 f_2\}$ condense on the boundary

Summarizing the boundaries and the condensates on the respective boundaries :

	Boundaries (Subgroups and 2-cocycles)	Condensate on the boundary
1	Trivial Subgroup,	$1 \oplus e_2 \oplus e_1 \oplus e_1 e_2$
2	$\{(), (3,4)\}$,	$1 \oplus e_1 \oplus m_2 \oplus e_1 m_2$
3	$\{(), (1,2)\}$,	$1 \oplus e_2 \oplus m_1 \oplus m_1 e_2$
4	$\{(), (1,2)(3,4)\}$	$1 \oplus e_1 e_2 \oplus m_1 m_2 \oplus f_1 f_2$
5	Group itself, Trivial cocycle	$1 \oplus m_2 \oplus m_1 \oplus m_1 m_2$
6	Group itself, Non trivial cocycle	$1 \oplus e_1 m_2 \oplus m_1 e_2 \oplus f_1 f_2$

There are few ways on relating the excitations in the unfolded case to the condensates on the boundary:

1. Considering the boundary as a infinite source of excitations
2. Observing the tensor split of characters at the boundary of the folded case.

Considering the boundary as a infinite source of excitations :
Consider the boundary of Trivial Subgroup,

$$e_1 \otimes (1 \oplus e_2 \oplus e_1 \oplus e_1 e_2) \text{ gives } (e_1 \oplus e_1 e_2 \oplus 1 \oplus e_2),$$

this implies e_1 condenses on the domain wall, reflects from the domain wall with e_2 on the other side, is identified with e_2 on the other side of the domain wall, pass through the domain wall as e_1.

Consider the case of $\{(), (3,4)\}$ as a boundary condition,

$$e_1 \otimes (1 \oplus e_1 \oplus m_2 \oplus e_1 m_2) \text{ gives } (e_1 \oplus 1 \oplus e_1 m_2 \oplus m_2)$$

Here the viewpoint is though there is a map that takes e_1 to m_2, it is that e_1 condenses and m_2 condense at the respective boundary. The actual map is then something on the lines of :

$$e_1 \longrightarrow 1 \leftrightarrow 1 \longleftarrow m_2$$

Observing the split of characters at the boundary of the folded case :
In the above listing, the excitations have been written as a direct sum of coupled excitations with the indexing referring to the groups involved in the direct product. In the unfolded case, the direct product can be seen as two lattices separated by a boundary. Therefore, the map is observing the terms in direct sum. For example, consider the case of group as subgroup with non-trivial cocycle, the condensate is given by
$1 \oplus e_1 m_2 \oplus m_1 e_2 \oplus f_1 f_2$ implying e_1 is mapped to m_2.

5.3 Gapped Boundaries of $S_3 \times S_3$ with non-trivial cocycles

This section aims at establishing the isomorphic map between $F_3^+ \rtimes F_3^\times$ and S_3 leading to the identification of explicit 2-cocycles of the group cohomology $H^2(S_3^2, U(1))$.

5.3.1 Map between $F_3^+ \rtimes F_3^\times$ and S_3

Let F_q be the finite field of cardinality q. Let F_q^+ and $F_q^\times$ be the additive and multiplicative groups of the finite field F_q. The semi-direct product of these groups, $F_q^+ \rtimes F_q^\times$, is given by the pair (a, α) where $a \in F_q^+$, $\alpha \in F_q^\times$, the product being defined as $(a, \alpha)(a^{'}, \alpha^{'})$ $= (a + \alpha a^{'}, \alpha\alpha^{'})$, the identity is given by (0,1) and the inverse by $(-\alpha^{-1}a, \alpha^{-1})$.

The elements of $F_3^+ \rtimes F_3^\times$ are given by $\{(0,1),(1,1),(2,1),(0,2),(1,2),(2,2)\}$ and that of S_3 are given by $\{e,(1,2),(2,3),(1,3),(1,2,3),(1,3,2)\}$. The isomorphism map f : $F_3^+ \rtimes F_3^\times \longrightarrow S_3$ is given by the following :

$$f(a,\alpha) = \begin{cases} e & \text{if } (a,\alpha) = (0,1) \\ 3 - cycle \ \ of \ \ S_3 & \text{if } \alpha = 1 \\ 2 - cycle \ \ of \ \ S_3 & \text{if } \alpha = 2 \end{cases}$$

One of the explicit map can be given as following :

$$\{(0,1),(0,2),(1,2),(2,2),(1,1),(2,1)\} \longmapsto \{e,(1,2),(1,3),(2,3),(1,3,2),(1,2,3)\}$$

5.3.2 Explicit cocycles for subgroups of $S_3 \times S_3$ leading to construction of Character

Define $U = \{((a_1,\alpha),(a_2,\alpha^{-1})), a_1, a_2 \in F_3^+, \alpha \in F_3^\times\}$. The explicit cocycle $\phi(g,h)$ is given by

$$\phi(g,h) = \omega^{tr_p(\alpha a_2 b_1)}$$

where $tr_p : F_q \longrightarrow F_p$, where p is the characteristic of F_q, i.e., $tr_p(a)$ is given by the trace of the linear map $x \longrightarrow ax$, $g = ((a_1,\alpha),(a_2,\alpha^{-1}))$, $h = ((b_1,\beta),(b_2,\beta^{-1}))$.

The character $\chi_{(K,\phi)}$ for a subgroup($K \subset G$) with cocycle(ϕ), which generalizes the Type-I boundaries is given by

$$\chi_{(K,\phi)}(gh^*) = \frac{1}{|K|}\delta_{hg,gh}\sum_{x\in G}\delta_{xgx^{-1}\in K}\delta_{xhx^{-1}\in K}\phi(xgx^{-1}|xhx^{-1})$$

where $\phi(k|l) = \phi(k,l)\phi(klk^{-1},k)^{-1}$

The character using U and the explicit cocycle construction reduces to

$$\chi_{(U,\phi)}(gh^*) = \begin{cases} \delta_{gh,hg}\delta_{g,h\in U}(q-1) & \text{if } \alpha \neq 1 or \beta \neq 1, \\ \delta_{gh,hg}\delta_{g,h\in U}\delta_{a_1b_2,a_2b_1}(q-1) - \delta_{a_1b_2\neq a_2b_1} & \text{if } \alpha=\beta=1 \end{cases}$$

for $g = ((a_1,\alpha),(a_2,\alpha^{-1})), h = ((b_1,\beta),(b_2,\beta^{-1}))$

We observe that the character is dependent on the subgroup but not explicity on the cocycle, though the dependence is reflected by the constraints on α, β.

Though U defined, is not ismorphic to the subsets of $S_3 \times S_3$, in the sense subsets of $(F_3^+ \rtimes F_3^\times)^2$ are not in ismorphism to U. U is just a special construction, and therefore the character is also for this special construction.

5.3.3 Condesations for a choice from U

Consider the following subset of $S_3 \times S_3$:
$\{e,(2,3)(2,3),(1,2)(1,2),(1,2,3)(1,2,3),(1,3,2)(1,3,2),(1,3)(1,3)\}$ which is equivalent $\{(0,1)(0,1),(0,2)(0,2),(1,1)(1,1),(1,2)(1,2),(2,1)(2,1),(2,2)(2,2)\}$. This is an element of U and hence the cocycle construction holds for this this choice. To compute the condesations on the boundary, we use use the explicit character construction and the equivalent character excitation

5.3.3.1 Using the explicit character construction and the equivalent character excitation

```
Finite field  Finite Field of size 3


elements of semi_direct_product
[[0, 1], [0, 2], [1, 1], [1, 2], [2, 1], [2, 2]]


subgroup
[[[0, 1], [0, 1]], [[0, 2], [0, 2]], [[1, 1], [1, 1]],
```

```
 [[1, 2], [1, 2]], [[2, 1], [2, 1]], [[2, 2], [2, 2]]]


Character
[[0, 1], [0, 1]] [[0, 1], [0, 1]] 2
[[0, 1], [0, 1]] [[0, 2], [0, 2]] 2
[[0, 1], [0, 1]] [[1, 1], [1, 1]] 2
[[0, 1], [0, 1]] [[1, 2], [1, 2]] 2
[[0, 1], [0, 1]] [[2, 1], [2, 1]] 2
[[0, 1], [0, 1]] [[2, 2], [2, 2]] 2
[[0, 2], [0, 2]] [[0, 1], [0, 1]] 2
[[0, 2], [0, 2]] [[0, 2], [0, 2]] 2
[[0, 2], [0, 2]] [[1, 1], [1, 1]] 0
[[0, 2], [0, 2]] [[1, 2], [1, 2]] 0
[[0, 2], [0, 2]] [[2, 1], [2, 1]] 0
[[0, 2], [0, 2]] [[2, 2], [2, 2]] 0
[[1, 1], [1, 1]] [[0, 1], [0, 1]] 2
[[1, 1], [1, 1]] [[0, 2], [0, 2]] 0
[[1, 1], [1, 1]] [[1, 1], [1, 1]] 2
[[1, 1], [1, 1]] [[1, 2], [1, 2]] 0
[[1, 1], [1, 1]] [[2, 1], [2, 1]] 2
[[1, 1], [1, 1]] [[2, 2], [2, 2]] 0
[[1, 2], [1, 2]] [[0, 1], [0, 1]] 2
[[1, 2], [1, 2]] [[0, 2], [0, 2]] 0
[[1, 2], [1, 2]] [[1, 1], [1, 1]] 0
[[1, 2], [1, 2]] [[1, 2], [1, 2]] 2
[[1, 2], [1, 2]] [[2, 1], [2, 1]] 0
[[1, 2], [1, 2]] [[2, 2], [2, 2]] 0
[[2, 1], [2, 1]] [[0, 1], [0, 1]] 2
[[2, 1], [2, 1]] [[0, 2], [0, 2]] 0
[[2, 1], [2, 1]] [[1, 1], [1, 1]] 2
[[2, 1], [2, 1]] [[1, 2], [1, 2]] 0
[[2, 1], [2, 1]] [[2, 1], [2, 1]] 2
[[2, 1], [2, 1]] [[2, 2], [2, 2]] 0
```

```
[[2, 2], [2, 2]] [[0, 1], [0, 1]] 2
[[2, 2], [2, 2]] [[0, 2], [0, 2]] 0
[[2, 2], [2, 2]] [[1, 1], [1, 1]] 0
[[2, 2], [2, 2]] [[1, 2], [1, 2]] 0
[[2, 2], [2, 2]] [[2, 1], [2, 1]] 0
[[2, 2], [2, 2]] [[2, 2], [2, 2]] 2

# listing only the non-zero trace terms
conjugacy class  [ () ]
() () ()
(2,3)(5,6) () (2,3)(5,6)
(1,2)(4,5) () (1,2)(4,5)
(1,2,3)(4,5,6) () (1,2,3)(4,5,6)
(1,3,2)(4,6,5) () (1,3,2)(4,6,5)
(1,3)(4,6) () (1,3)(4,6)

conjugacy class  [ (2,3)(5,6), (2,3)(4,5), (2,3)(4,6), (1,2)
   (5,6), (1,2)(4,5), (1,2)(4,6),
  (1,3)(5,6), (1,3)(4,5), (1,3)(4,6) ]
() (2,3)(5,6) ()
() (1,2)(4,5) ()
() (1,3)(4,6) ()
(2,3)(5,6) (2,3)(5,6) (2,3)(5,6)
(1,2)(4,5) (1,2)(4,5) (2,3)(5,6)
(1,3)(4,6) (1,3)(4,6) (2,3)(5,6)

conjugacy class  [ (1,2,3)(4,5,6), (1,2,3)(4,6,5), (1,3,2)
   (4,5,6), (1,3,2)(4,6,5) ]
() (1,2,3)(4,5,6) ()
() (1,3,2)(4,6,5) ()
(1,2,3)(4,5,6) (1,2,3)(4,5,6) (1,2,3)(4,5,6)
(1,2,3)(4,5,6) (1,3,2)(4,6,5) (1,3,2)(4,6,5)
(1,3,2)(4,6,5) (1,2,3)(4,5,6) (1,3,2)(4,6,5)
```

```
(1,3,2)(4,6,5)  (1,3,2)(4,6,5)  (1,2,3)(4,5,6)
```

The following are the terms of the sum

For conjugacy class : [()]

Non-zero character terms of $S_3 \times S_3$	Equivalence in $F_3^+ \rtimes F_3^\times$	Trace term	Corresponding inner product term
() ()	(0,1)(0,1) (0,1)(0,1)	()	$2tr_{\pi_i}()$
(2,3)(5,6) ()	(2,2)(2,2) (0,1)(0,1)	(2,3)(5,6)	$2tr_{\pi_i}(2,3)(5,6)$
(1,2)(4,5) ()	(0,2)(0,2) (0,1)(0,1)	(1,2)(4,5)	$2tr_{\pi_i}(1,2)(4,5)$
(1,2,3)(4,5,6) ()	(2,1)(2,1) (0,1)(0,1)	(1,2,3)(4,5,6)	$2tr_{\pi_i}(1,2,3)(4,5,6)$
(1,3,2)(4,6,5) ()	(1,1)(1,1) (0,1)(0,1)	(1,3,2)(4,6,5)	$2tr_{\pi_i}(1,3,2)(4,6,5)$
(1,3)(4,6) ()	(1,2)(1,2) (0,1)(0,1)	(1,3)(4,6)	$2tr_{\pi_i}(1,3)(4,6)$

The ICC of (e,e) of $S_3 \times S_3$ is given by ICC of e of S_3 and ICC of e of S_3, say are given by A_1, B_1, C_1 and A_2, B_2, C_2 respectively.

The inner product terms are given by

$$2tr_{\pi_i}() + 2tr_{\pi_i}(2,3)(5,6) + 2tr_{\pi_i}(1,2)(4,5) + 2tr_{\pi_i}(1,2,3)(4,5,6) + 2tr_{\pi_i}(1,3,2)(4,6,5) + 2tr_{\pi_i}(1,3)(4,6)$$

This implies the condensates on the boundary are A_1A_2, B_1B_2, C_1C_2

For conjugacy class [(2,3)(5,6), (2,3)(4,5), (2,3)(4,6),
(1,2)(5,6), (1,2)(4,5), (1,2)(4,6), (1,3)(5,6), (1,3)(4,5), (1,3)(4,6)]

Non-zero character terms of $S_3 \times S_3$	Equivalence in $F_3^+ \rtimes F_3^\times$	Trace term	Corresponding inner product term
() (2,3)(5,6)	(0,1)(0,1) (2,2)(2,2)	()	$2tr_{\pi_i}()$
() (1,2)(4,5)	(0,1)(0,1) (0,2)(0,2)	()	$2tr_{\pi_i}()$
() (1,3)(4,6)	(0,1)(0,1) (1,2)(1,2)	()	$2tr_{\pi_i}()$
(2,3)(5,6)	(2,2)(2,2) (2,2)(2,2)	(2,3)(5,6)	$2tr_{\pi_i}(2,3)(5,6)$
(2,3)(5,6)			
(1,2)(4,5)	(0,2)(0,2) (0,2)(0,2)	(2,3)(5,6)	$2tr_{\pi_i}(2,3)(5,6)$
(1,2)(4,5)			
(1,3)(4,6)	(1,2)(1,2) (1,2)(1,2)	(2,3)(5,6)	$2tr_{\pi_i}(2,3)(5,6)$
(1,3)(4,6)			

The ICC of $(\bar{\tau}, \bar{\tau})$ of $S_3 \times S_3$ is given by ICC of τ of S_3 and ICC of τ of S_3, say are given by D_1, E_1 and D_2, E_2 respectively.

The inner product terms are given by

$$6tr_{\pi_i}() + 6tr_{\pi_i}(2,3)(5,6)$$

D_1D_2, E_1, E_2 condense on the boundary.

For conjugacy class [(1,2,3)(4,5,6), (1,2,3)(4,6,5), (1,3,2)(4,5,6), (1,3,2)(4,6,5)]

Non-zero character terms of $S_3 \times S_3$	Equivalence in $F_3^+ \rtimes F_3^\times$	Trace term	Corresponding inner product term
() (1,2,3)(4,5,6)	(0,1)(0,1) (2,1)(2,1)	()	$2tr_{\pi_i}()$
() (1,3,2)(4,6,5)	(0,1)(0,1) (1,1)(1,1)	()	$2tr_{\pi_i}()$
(1,2,3)(4,5,6)	(2,1)(2,1) (2,1)(2,1)	(1,2,3)(4,5,6)	$2tr_{\pi_i}(1,2,3)(4,5,6)$
(1,2,3)(4,5,6)			
(1,2,3)(4,5,6)	(2,1)(2,1) (1,1)(1,1)	(1,3,2)(4,6,5)	$2tr_{\pi_i}(1,3,2)(4,6,5)$
(1,3,2)(4,6,5)			
(1,3,2)(4,6,5)	(1,1)(1,1) (2,1)(2,1)	(1,3,2)(4,6,5)	$2tr_{\pi_i}(1,3,2)(4,6,5)$
(1,2,3)(4,5,6)			
(1,3,2)(4,6,5)	(1,1)(1,1) (1,1)(1,1)	(1,2,3)(4,5,6)	$2tr_{\pi_i}(1,2,3)(4,5,6)$
(1,3,2)(4,6,5)			

The ICC of $(\bar{\sigma}, \bar{\sigma})$ of $S_3 \times S_3$ is given by ICC of σ of S_3 and ICC of σ of S_3, say are given by F_1, G_1, H_1 and F_2, G_2, H_2 respectively.

The inner product terms are given by

$$2tr_{\pi_i}() + 2tr_{\pi_i}(1,2,3)(4,5,6) + 2tr_{\pi_i}(1,3,2)(4,6,5)$$

G_1H_2, H_1G_2 condense on the boundary.

Similar analysis can be done for the subgroup of U = [[0,1],[0,1]], [[0,1],[2,1]], [[0,1],[1,1]], [[2,2],[2,2]], [[2,2],[0,2]], [[2,2],[1,2]], [[0,2],[2,2]], [[0,2],[0,2]], [[0,2],[1,2]], [[2,1],[0,1]], [[2,1],[2,1]], [[2,1],[1,1]], [[1,1],[0,1]], [[1,1],[2,1]], [[1,1],[1,1]], [[1,2],[2,2]], [[1,2],[0,2]], [[1,2],[1,2]] which has C_1F_2 and C_2F_1 condensing on the boundary.

Similar construction can be studied for any arbitrary group $G_1 \times G_2$.The motivation for constructing various boundaries for the non-abelain case of S_3 is to relate to the relevant SPT phases which when gauged give rise to gapped domain walls in Quantum Double Models [12][13] [14][15][16][17]. The non trivial Hamiltonian of a SPT phase is given by $H_1 = U_{v_d} H_0 U_{v_d}^{\dagger}$ where H_0 is the trivial Hamiltonian and U_{v_d} is a unitary transformation. The motivation was to construct a explicit unitary tranformation given a subgroup and non-trivial cocycle for a non-abelian group, giving a direct relationship between gapped boundaries, quantum gates and SPT phases.

Chapter 6

Boundary conditions of String-Net Fibonacci Model

The construction of ribbon operator in the Quantum Double Model was done by the construction of certain commutation relationships. This section aims at presenting certain attempts to mimic the same for Levin-Wen String Net Model with the UTC being the Fibonacci Tensor Category with objects $\{1, \tau\}$ on a two site lattice on a torus. A generalization is presented for the commutation relationship between W_e^{pq} which is the string operator and Q_v^q.

6.1 Commutation conditions between the string operator $W_e^{p',q'}$ and the Q_v^q

Observing the equation 62 from [8], the relation between Q_v^q and an edge decorated with the charge by q_1 is given by $\delta_{q_1,q}$. Starting with a state already decorated with charges p, q^* on an edge. The action of $Q_v^{q_1}$ followed by $W_e^{p^{'},q^{'}}$ is equivalent to δ_{q^*,q_1} and using certain F-moves the final state would have a fusion of q^* and q^1 where the inital charge decoration was q^*. Similary the action of $W_e^{p^{'},q^{'}}$ followed by $Q_v^{q_1}$ results in the same state but accompanied by a different condition given by $\delta_{q_1,\sum q^* \oplus q'}$ where $q^* \oplus q^{'}$ is the fusion rule between q^* and $q^{'}$. Therefore,

$$Q_v^{q_1} W_e^{p^{'},q^{'}} = \delta_{q_1,\sum q^* \oplus q'} \div \delta_{q^*,q_1} W_e^{p^{'},q^{'}} Q_v^{q_1} \tag{6.1}$$

6.2 Commutation conditions between the string operator $W_e^{p',q'}$ and B_p^s

In this case, we present a procedure to compute the commutation relationship between the string operator W and face operator B on a two site lattice on a torus. The initial state is decorated with the charge edges p, q^*. The BW and WB operators are as follows :

6.2.1 WB operator

$R^{4_r}_{q^*4_1}\ (B^s_{p\textcircled{1}} = F^{4_r^{'}*56^*}_{s6^{'}*5^{'}}\ F^{345^*}_{s5^{'}*4^{'}} F^{p^*4_14^*}_{s4^{'}*4_1^{'}})\ (B^s_{p\textcircled{2}} = F^{q^*4_r^{'}4_1^{'*}}_{s4_1^{''}*4_r^{''}}\ F^{5^{'}6^{'}*4_r^{'*}}_{s4_r^{''}*6^{''*}})\ (B^s_{p\textcircled{3}} = F^{12^*6^{''}}_{s6^{'''}2^{'*}} F^{3^*1^*2}_{s2^{'}1^{'*}})$
$(B^s_{p\textcircled{4}} = F^{2^{'}*6^{'''}1^{'}}_{s1^{''}6^{''''}})\ R^{4_r^{''}}_{q^*4_1^{''}} \sum_{4_r^{''''}} F^{q^*4_1^{''*}4_r^{'''}}_{5^{'}6^{''''}*4_r^{''''}} F^{q^*4_r^{''''}6^{''''*}}_{6^{''''}*0q} \sum_{4^{''*}} F^{p^*4_1^{''}4^{'*}}_{5^{'*}34^{''*}}\ F^{p^*4^{''*}3}_{30p} \sum_{4_2} \frac{v_{4_2}}{v_{4_1^{''}}} z_{q^{'}4_2p^{'}4_1^{''}}$
$\sum_{4_1^{'''},4_r^{''''}} F^{q^{'}*4_2^*4_1^{''}}_{5^{'}4_r^{''''}*4_1^{'''}}\ F^{q^{'}*4_1^{'''}4_r^{''''*}}_{6^{''''}*q^*4_r^{''''}} \sum_{4_1^{''''},4^{'''}} F^{p^{'}*4_2^{''}4_1^{''*}}_{5^{'}*4^{''}4_1^{''''}}\ F^{3p^*4^{''*}}_{p^{'}*4_1^{''''}4^{'''}} \sum_{4_1^{''''}} F^{4_2^*5^{'}4_1^{'''*}}_{6^{''''}*4_r^{''''}*4_1^{''''}} \sum_{4_1^{''''''}} F^{34^{'''}*4_1^{''''}}_{4_25^{'}*4_1^{''''''}}$

6.2.2 BW operator

$\sum_{4^{'}} F^{q^*4_1^*4}_{56^*4^{'}} F^{q^*4^{'}6^*}_{6^*0q} \sum_{4^{'}} F^{q^*4_1^*4_q}_{56^*4^{'}} F^{q^*4^{'}6^*}_{6^*0q} \sum_{4^{''*}} F^{p^*4_14^*}_{5^*34^{''*}} F^{303^*}_{p^*4^{''*}p^*} \sum_{4^{''*}} F^{p^*4_14_p^*}_{5^*34^{''*}} F^{303^*}_{p^*4^{''*}p^*} \sum_{4_2} \frac{v_{4_2}}{v_{4_1}} z_{4_1q^{'}4_2p^{'}} =$
$\sum_{4_2} \frac{v_{4_2}}{v_{4_1}} R^{4_1}_{14_2}$
(only for Fibonaaci labels, Eq : 93) and this 1 in the lower indices is charge labelling not the edge labelled 1
$\sum_{4_1^{'},4^{'''}} F^{q^{'}*4_2^*4_1}_{54^{'}*4_1^{'}} F^{q^{'}*4_1^{'}4^{'*}}_{6^*q^*4^{'''}} \sum_{4_1^{''},4^{''''}} F^{p^{'}*4_24_1^*}_{5^*4^{''}4_1^{''*}} F^{3p^*4^{''*}}_{p^{'}*4_1^{''}*4^{''''*}} \sum_{4_1^{'''}} F^{4_2^*54_1^{'*}}_{6^*4^{'''}4_1^{'''}} \sum_{4_1^{''''}} F^{34^{'''}4_1^{''*}}_{4_25^*4_1^{''''}} R^{4_1^{'''}}_{4^{'''}*4_2^*}$
$(B^s_{p\textcircled{1}} = F^{4^{''''}*4_24_1^{''''}}_{s4_1^{''''''}4_2^{'}}\ F^{34_1^{''''}*5^*}_{s5^{'}*4_1^{''''''}*}\ F^{4_1^{''''''}*56^*}_{s6^{'}*5^{'}})\ (B^s_{p\textcircled{2}} = F^{4^{'''}*4_1^{''''}4_2^{'*}}_{s4_2^{''}*4_1^{''''''}}\ F^{5^{'}6^{'}*4_1^{''''''*}}_{s4_1^{''''''}*6^{''*}})\ (B^s_{p\textcircled{3}} = F^{12^*6^{''}}_{s6^{'''}2^{'*}} F^{3^*1^*2}_{s2^{'}1^{'*}})$
$(B^s_{p\textcircled{4}} = F^{2^{'}*6^{'''}1^{'}}_{s1^{''}6^{''''}})\ (R^{4_1^{''''''}}_{4^{'''}*4_2^{''*}})$

In order to compute the commutation relationship, the sum over the F-symbols has to be done, which is computationally pretty costly. This is one of the methods that can be employed but not completely efficient.

This section aimed at presenting the commutation relationships between the string operator, which is the ribbon operator equivalent, and the vertex, face operators.

Chapter 7

Summary and Future Directions

7.1 Summary

To summarize, we started with a brief introduction to category theory and some mathematical structures like Algebras, Bialgebras and Hopf Algebras which were used to construct the Drinfeld Double of a group. We then introduced the lattice models which are used to classify the topological phases of matter. Kitaev Quantum Double Models were introduced along with some properties of the system like ground state construction, ribbon operator construction leading to detection of excitations given a particular group. Construction of ribbon operators in a Quantum Double with boundaries leading to detection of condensation of excitations on boundaries is presented. SageMath [18] has been used to compute the ribbon operators on lattice with boundaries, excitations which condense on the boundary, the results for S_3 have been presented in thesis and for any generalized group the methods to compute the same have been presented in the Appendix A. An introduction to Levin-Wen models with and without boundaries, with a brief introduction to excitation detection and string operator that is the ribbon operator equivalent have been presented. Statements relating excitations in the bulk, excitations on the boundary and excitations in the condensed phase have been presented in one step as well as two step condensation. In the end, an attempt has been made to show Quantum Doubles as a subclass of Levin-Wen models and various results have been verified in the case of the Toric Code and $D(S_3)$.

7.2 Future Directions

Some of the following ideas require further work :

- Computing the commutation relationship equivalent of ribbon operators for string operators. Solving for string operators in the presence of boundary using the commutation relationship. For the Q_v operator, the result has been computed but B_p operator is still a work in progress. Once the string operators in a lattice with boundaries are computed, the next step is to compute the ground state which would lead to the computation of topological entanglement entropy.
- Construction of ribbon operators in higher dimensions [12].
- Construction of boundary conditions for Twisted Quantum Double models in 2D [19, 20] and 3D [21], and extending the idea of ribbon operators for these models with boundaries.
- In the case of $D(S_3)$, construction of condensable algebras play an important role as these would help in understanding the computation of boundary excitations, condensed phase and excitations which condense on the boundary. Also topology of quantum algorithms uses the same structures, so the study of these using the above would form an interesting case [22, 23].
- Topological Entanglement Entropy in the presence of ribbon operators in the case of $D(S_3)$. The entanglement entropy for a $G_1 \times G_2$ model with boundaries has been computed.
- Given the center of a UTC which is a MTC, one can find many UTC's which are equivalent upto Morita equivalence. Given a UTC, one can construct the boundary labels as left modules of the UTC. Therefore, given a MTC, is it possible to predict the boundary labels.
- Understanding of co-dimension 2 boundaries and application to the case of $D(S_3)$.
- The ribbon operators computed for S_3 do not provide the required insight on splitting of excitation on boundary. In the sense the action of the ribbon operator on a cylinderical lattice connecting boundaries at the top and bottom, is equivalent

to the horizontal surface action. The construction of ribbon operator with vertical surface action might provide an insight into the splitting of excitation on boundary.

Appendix A

Functions used to calculate various properties of Quantum Double Models

Quantum Double Models using SageMath

0.1 Simple Mathematics using SageMath

```
In [6]: # Finding the gcd of two numbers
        gcd?
```

```
In [7]: gcd(25,30)
```

```
Out[7]: 5
```

```
In [8]: factor(625)
```

```
Out[8]: 5^4
```

```
In [9]: factor(2435)
```

```
Out[9]: 5 * 487
```

```
In [11]: M = matrix([[1,2,4],[1,3,5],[5,4,2]]);M
```

```
Out[11]: [1 2 4]
         [1 3 5]
         [5 4 2]
```

```
In [12]: M.inverse()
```

```
Out[12]: [   7/6     -1    1/6]
         [-23/12    3/2   1/12]
         [ 11/12   -1/2  -1/12]
```

```
In [13]: M.eigenvalues()
```

```
Out[13]: [-3.583940313126412?, 0.3631206519422502?, 9.220819661184l7?]
```

0.2 Group Theory in SageMath

```
In [15]: Z2 = SymmetricGroup(2); Z2
```

```
Out[15]: Symmetric group of order 2! as a permutation group
```

```
In [16]: S3 = SymmetricGroup(3); S3
```

```
Out[16]: Symmetric group of order 3! as a permutation group
```

```
In [17]: Z2.is_subgroup(S3)
```

```
Out[17]: True
```

```
In [18]: S3.is_cyclic()
```

```
Out[18]: False
```

```
In [19]: Z2.is_abelian()
```

```
Out[19]: True
```

0.3 Excitations, Ribbon operators, Ground states in Quantum Double Models

Defining different models - Quantum Double of Z2, S3, D4

```
In [1]: QDM_Toric = SymmetricGroup(2)
        QDM_Toric

Out[1]: Symmetric group of order 2! as a permutation group

In [2]: QDM_S3 = SymmetricGroup(3)
        QDM_S3

Out[2]: Symmetric group of order 3! as a permutation group

In [3]: QDM_D4 = DihedralGroup(4)
        QDM_D4

Out[3]: Dihedral group of order 8 as a permutation group
```

Developing the machinery to compute the number of excitations.

1. **Computing the centralizers of the conjugacy class of the group.**

```
In [4]: def centralizer_conjugacy_class_QDM_generic(QDM_group):
            cent_QDM_group = []
            for conj_class in QDM_group.conjugacy_classes():
                centralizer = QDM_group.centralizer(conj_class.an_element())
                cent_QDM_group.append(centralizer)
            return cent_QDM_group

In [5]: cent_toric = centralizer_conjugacy_class_QDM_generic(QDM_Toric)
        cent_toric

Out[5]: [Subgroup of (Symmetric group of order 2! as a permutation group) generated by [(1,2)],
         Subgroup of (Symmetric group of order 2! as a permutation group) generated by [(1,2)]]

In [6]: cent_s3 = centralizer_conjugacy_class_QDM_generic(QDM_S3)
        cent_s3

Out[6]: [Subgroup of (Symmetric group of order 3! as a permutation group) generated by [(2,3), (1,3)],
         Subgroup of (Symmetric group of order 3! as a permutation group) generated by [(1,2)],
         Subgroup of (Symmetric group of order 3! as a permutation group) generated by [(1,2,3)]]

In [7]: cent_d4 = centralizer_conjugacy_class_QDM_generic(QDM_D4)
        cent_d4

Out[7]: [Subgroup of (Dihedral group of order 8 as a permutation group) generated by [(1,2,3,4), (1,4)(
         Subgroup of (Dihedral group of order 8 as a permutation group) generated by [(2,4), (1,3)(2,4)]
         Subgroup of (Dihedral group of order 8 as a permutation group) generated by [(1,2)(3,4), (1,3)
         Subgroup of (Dihedral group of order 8 as a permutation group) generated by [(1,2,3,4), (1,3)(
         Subgroup of (Dihedral group of order 8 as a permutation group) generated by [(1,2,3,4), (1,4)(
```

2. The character table gives the trace of irreducible representations (but the trace is used at a later stage).

```
In [8]: def character_table_centralizers(centralizers_generic_group):
            char_table = []
            for subgroup in centralizers_generic_group:
                char_table.append(subgroup.character_table())
            return char_table
```

```
In [9]: cent_toric_centralizer_character = character_table_centralizers(cent_toric)
        cent_toric_centralizer_character
```

```
Out[9]: [
        [ 1 -1]  [ 1 -1]
        [ 1  1], [ 1  1]
        ]
```

```
In [10]: cent_s3_centralizer_character_table = character_table_centralizers(cent_s3)
         cent_s3_centralizer_character_table
```

```
Out[10]: [
         [ 1 -1  1]           [         1          1          1]
         [ 2  0 -1]  [ 1 -1]  [         1      zeta3 -zeta3 - 1]
         [ 1  1  1], [ 1  1], [         1 -zeta3 - 1      zeta3]
         ]
```

```
In [11]: cent_d4_centralizer_character_table = character_table_centralizers(cent_d4)
         cent_d4_centralizer_character_table
```

```
Out[11]: [
         [ 1  1  1  1  1]
         [ 1 -1 -1  1  1]  [ 1  1  1  1]  [ 1  1  1  1]
         [ 1 -1  1 -1  1]  [ 1 -1 -1  1]  [ 1 -1 -1  1]
         [ 1  1 -1 -1  1]  [ 1 -1  1 -1]  [ 1 -1  1 -1]
         [ 2  0  0  0 -2], [ 1  1 -1 -1], [ 1  1 -1 -1],

                                          [ 1  1  1  1  1]
         [     1      1      1      1]    [ 1 -1 -1  1  1]
         [     1     -1      1     -1]    [ 1 -1  1 -1  1]
         [     1 -zeta4     -1  zeta4]    [ 1  1 -1 -1  1]
         [     1  zeta4     -1 -zeta4], [ 2  0  0  0 -2]
         ]
```

3. Computing the number of excitations by counting the number of rows in the character table.

```
In [12]: def excitations_count(QDM_group):
             count = 0
             generic_centralizer_character_table = character_table_centralizers(centralizer_conjugacy_c
             for char_table in generic_centralizer_character_table:
                 count += char_table.nrows()
             return count
```

```
In [13]: QDM_toric_excitations = excitations_count(QDM_Toric)
         QDM_toric_excitations
```

```
Out[13]: 4
```

```
In [14]: QDM_S3_excitations = excitations_count(QDM_S3)
         QDM_S3_excitations
```

```
Out[14]: 8
```

```
In [15]: QDM_D4_excitations = excitations_count(QDM_D4)
         QDM_D4_excitations
```

```
Out[15]: 22
```

Developing the machinery to compute the excitations that condense on a given boundary

1. **Computing the character related to the irreducible representation of the group.**

```
In [16]: def character_excitation(G, conjugacy_class, g, h):
             k_h = 0
             for g_1 in G:
                 if h*g_1 == g_1*conjugacy_class.an_element():
                     k_h = g_1
                     break
             if g*h == h*g and k_h != 0:
                 return k_h^-1*g*k_h
             else:
                 return 0
```

2. **Computing the character related to a particular boundary.**

```
In [17]: def character_subgroup(G, subgroup, g, h):
             sum = 0
             if h*g == g*h:
                 for g_1 in G:
                     if g_1*g*g_1^-1 in subgroup and g_1*h*g_1^-1 in subgroup:
                         sum = sum + 1
             return sum/len(subgroup)
```

3. **Computing the inner product terms of the above characters.**

```
In [18]: def inner_product_of_characters(QDM_group, subgroup, conjugacy_class):
             inner_product_terms = []
             for g in QDM_group:
                 for h in QDM_group:
                     if character_subgroup(QDM_group, subgroup, g, h) != 0 and character_excitation(QDM
                         inner_product_terms.append([character_subgroup(QDM_group, subgroup, g, h), cha
             return inner_product_terms
```

```
In [19]: inner_product_of_characters(QDM_S3, QDM_S3.subgroups()[5], QDM_S3.conjugacy_classes()[0])
```

```
Out[19]: [[1, ()], [1, (1,2)], [1, (1,2,3)], [1, (1,3,2)], [1, (2,3)], [1, (1,3)]]
```

$1 * tr_{\pi_i}(e) + 1 * tr_{\pi_i}(1,2) + 1 * tr_{\pi_i}(1,2,3) + 1 * tr_{\pi_i}(1,3,2) + 1 * tr_{\pi_i}(2,3) + 1 * tr_{\pi_i}(1,3)$

From the character table for S_3, and labelling each excitation

$\{e\}$	$\{\tau\}$	$\{\sigma\}$				
1	-1	1	$->$	tr_{π_2}	$->$	B
2	0	-1	$->$	tr_{π_3}	$->$	C
1	1	1	$->$	tr_{π_1}	$->$	A

Therefore A condenses on the boundary as the inner product is greater than zero, the others go to zero.

```
In [20]: inner_product_of_characters(QDM_S3, QDM_S3.subgroups()[5], QDM_S3.conjugacy_classes()[1])

Out[20]: [[1, ()], [1, ()], [1, ()], [1, (1,2)], [1, (1,2)], [1, (1,2)]]
```

$3 * tr_{\pi_i}(e) + 3 * tr_{\pi_i}(1,2)$
From the character table for Z_2, and labelling each excitation
$1 \quad -1 \quad tr_{\pi_2} \quad -> \quad E$
$1 \quad 1 \quad tr_{\pi_1} \quad -> \quad D$
Therefore D condenses on the boundary as the inner product is greater than zero, the others go to zero.

```
In [21]: inner_product_of_characters(QDM_S3, QDM_S3.subgroups()[5], QDM_S3.conjugacy_classes()[2])

Out[21]: [[1, ()], [1, ()], [1, (1,2,3)], [1, (1,3,2)], [1, (1,3,2)], [1, (1,2,3)]]
```

$2 * tr_{\pi_i}(e) + 2 * tr_{\pi_i}(1,2,3) + 2 * tr_{\pi_i}(1,3,2)$
From the character table for Z_3, and labelling each excitation
$1 \quad 1 \quad 1 \quad tr_{\pi_1} \quad -> \quad F$
$1 \quad zeta3 \quad -zeta3-1 \quad tr_{\pi_2} \quad -> \quad G$
$1 \quad -zeta3-1 \quad zeta3 \quad tr_{\pi_3} \quad -> \quad H$
Therefore F condenses on the boundary as the inner product is greater than zero, the others go to zero.
Hence, for the subgroup $K = G$, the excitations A, D, F condense on the boundary.

Similarly varying the boundaries (different subgroups) and using the inner product, the excitations which condense on the boundary can be determined.

```
In [22]: def boundary_condensates(QDM_group, QDM_subgroup):
             total_inner_product_terms = []
             for conj_class in QDM_group.conjugacy_classes():
                 total_inner_product_terms.append(inner_product_of_characters(QDM_group, QDM_subgroup, 
             return total_inner_product_terms
```

Boundary condensates for the boundary indexed by $\{e, \tau\}$

```
In [23]: boundary_condensates(QDM_S3, QDM_S3.subgroups()[1])

Out[23]: [[[3, ()], [1, (1,2)], [1, (2,3)], [1, (1,3)]],
          [[1, ()], [1, ()], [1, ()], [1, (1,2)], [1, (1,2)], [1, (1,2)]],
          []]
```

Observing the character table list, A, C, D condense given the boundary is indexed by $\{e, \tau\}$

Construction of the ribbon operators for lattice with boundary Given that the boundary is given by the boundary (subgroup K), the ribbon operator with an excitation in the bulk and the condensate on the boundary is given by
$T^{(k,g)} = \Sigma_{l \in K} F^{(lkl^{-1}, gl^{-1})}$ where $k \in K, g \in G$
Fixing the subgroup $K = \{e, \tau\}, (\{e, (2,3)\}$ for example)

```
In [24]: K = QDM_S3.subgroups()[1];K

Out[24]: Subgroup of (Symmetric group of order 3! as a permutation group) generated by [(2,3)]
```

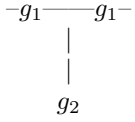
```
In [25]: def ribbon_operator_constructs(QDM_group, subgroup):
             ribbon_operator_terms = []
             for k in subgroup:
                 for g in QDM_group:
                     for l in subgroup:
                         ribbon_operator_terms.append([k,g,l*k*l^-1, l*g^-1])
             return ribbon_operator_terms
         ribbon_operator_constructs(QDM_S3, K)
```

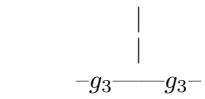
```
Out[25]: [[(), (), (), ()],
          [(), (), (), (2,3)],
          [(), (1,2), (), (1,2)],
          [(), (1,2), (), (1,2,3)],
          [(), (1,2,3), (), (1,3,2)],
          [(), (1,2,3), (), (1,3)],
          [(), (1,3,2), (), (1,2,3)],
          [(), (1,3,2), (), (1,2)],
          [(), (2,3), (), (2,3)],
          [(), (2,3), (), ()],
          [(), (1,3), (), (1,3)],
          [(), (1,3), (), (1,3,2)],
          [(2,3), (), (2,3), ()],
          [(2,3), (), (2,3), (2,3)],
          [(2,3), (1,2), (2,3), (1,2)],
          [(2,3), (1,2), (2,3), (1,2,3)],
          [(2,3), (1,2,3), (2,3), (1,3,2)],
          [(2,3), (1,2,3), (2,3), (1,3)],
          [(2,3), (1,3,2), (2,3), (1,2,3)],
          [(2,3), (1,3,2), (2,3), (1,2)],
          [(2,3), (2,3), (2,3), (2,3)],
          [(2,3), (2,3), (2,3), ()],
          [(2,3), (1,3), (2,3), (1,3)],
          [(2,3), (1,3), (2,3), (1,3,2)]]
```

$$
\begin{aligned}
T^{(e,e)} &= F^{(e,e)} + F^{(e,(2,3))},\\
T^{(e,(1,2))} &= F^{(e,(1,2))} + F^{(e,(1,2,3))},\\
T^{(e,(1,2,3))} &= F^{(e,1,3,2)} + F^{(e,(1,3))},\\
T^{((2,3),e)} &= F^{((2,3),e)} + F^{((2,3),(2,3))},\\
T^{((2,3),(1,2))} &= F^{((2,3),(1,2))} + F^{((2,3),(1,2,3))},\\
T^{((2,3),(1,2,3)} &= F^{((2,3),(1,3,2))} + F^{((2,3),(1,3))},
\end{aligned}
$$

Similarly for various boundaries, various ribbon operators connecting the bulk to the boundary can be generated. It is observed that for every boundary (every subgroup) there are 6 unique ribbon operators connecting the bulk to boundary in the case of S_3

Ground states with respect to different `T` operators on a cylinder with a single lattice (implying boundary on both sides of the lattice) The lattice looks in the following way :





Eigenstates of $\Pi\{\Sigma\ (vertex\ operators)\}\ (face\ operators)T$ are the ground states of the lattice with a ribbon operator. In the above lattice $g_1 and g_3$ are restricted to the subgroup (identified as boundary). There are three conditions to be satisfied, fixing the boundary to be $\{e, \tau\}$, due to the ribbon operators g_2 is restricted to $\{e, (2,3)\}$, due to the face operators the relationship between g_1, g_2, g_3 is as follows $g_3 g_2 g_1 g_2^{-1} = e$, and finally due to the vertex operators g_1, g_2, g_3 get mapped to $k_u g_1 k_u^{-1}, k_d g_2 k_u^{-1}, k_d g_3 k_d^{-1}$ respectively, where $k_u, k_d \in K$

```
In [26]: def ground_state_terms(g1, g2, g3, ku,  kd):
             return ku*g1*ku^-1, kd*g2*ku^-1, kd*g3*kd^-1
```

```
In [27]: def ground_state_sum(condition_set, subgroup):
             s = []
             for g2 in condition_set[1]:
                 for g3 in subgroup:
                     for g1 in subgroup:
                         if condition_set[0]*g3*g2*condition_set[0]*g1 == g2:
                             s.append((condition_set[0]*g1,g2,condition_set[0]*g3))
                             for i in subgroup:
                                 for j in subgroup:
                                     s.append([ground_state_terms(condition_set[0]*g1, g2, condition_se
             return s
```

Observing that $T^{(e,e)} = F^{(e,e)} + F^{(e,(2,3))}$ the condition set is that $g_2 \in \{e, (2,3)\}$ similarly to determine the other ground states the condition set is required

```
In [28]: ground_state_sum([QDM_S3[0],[QDM_S3[0], QDM_S3[4]]], QDM_S3.subgroups()[1])
```

```
Out[28]: [((), (), ()),
          [((), (), ())],
          [((), (2,3), ())],
          [((), (2,3), ())],
          [((), (), ())],
          ((2,3), (), (2,3)),
          [((2,3), (), (2,3))],
          [((2,3), (2,3), (2,3))],
          [((2,3), (2,3), (2,3))],
          [((2,3), (), (2,3))],
          (), (2,3), ()),
          [((), (2,3), ())],
          [((), (), ())],
          [((), (), ())],
          [((), (2,3), ())],
          ((2,3), (2,3), (2,3)),
          [((2,3), (2,3), (2,3))],
          [((2,3), (), (2,3))],
          [((2,3), (), (2,3))],
          [((2,3), (2,3), (2,3))]]
```

This implies for the operator $T^{(e,e)}$:

Possible initial configuration	*Ground state*
(e, e, e)	$2 * (e, e, e) + 2 * (e, (2,3), e)$
$((2,3), e, (2,3))$	$2 * ((2,3), e, (2,3)) + 2 * ((2,3), (2,3), (2,3))$
$(e, (2,3), e)$	$2 * (e, e, e) + 2 * (e, (2,3), e)$
$((2,3), (2,3), (2,3))$	$2 * ((2,3), e, (2,3)) + 2 * ((2,3), (2,3), (2,3))$

```
In [29]: ground_state_sum([QDM_S3[0],[QDM_S3[1], QDM_S3[2]]], QDM_S3.subgroups()[1])
```

```
Out[29]: [((), (1,2), ()),
          [((), (1,2), ())],
          [((), (1,2,3), ())],
          [((), (1,3,2), ())],
          [((), (1,3), ())],
          ((), (1,2,3), ()),
          [((), (1,2,3), ())],
          [((), (1,2), ())],
          [((), (1,3), ())],
          [((), (1,3,2), ())]]
```

This implies for the operator $T^{(e,(1,2))}$:

Possible initial configuration	*Ground state*
$(e,(1,2),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$
$(e,(1,2,3),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$

```
In [30]: ground_state_sum([QDM_S3[0],[QDM_S3[3], QDM_S3[5]]], QDM_S3.subgroups()[1])
```

```
Out[30]: [((), (1,3,2), ()),
          [((), (1,3,2), ())],
          [((), (1,3), ())],
          [((), (1,2), ())],
          [((), (1,2,3), ())],
          ((), (1,3), ()),
          [((), (1,3), ())],
          [((), (1,3,2), ())],
          [((), (1,2,3), ())],
          [((), (1,2), ())]]
```

This implies for the operator $T^{(e,(1,2,3))}$:

Possible initial configuration	*Ground state*
$(e,(1,3),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$
$(e,(1,3,2),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$

```
In [31]: ground_state_sum([QDM_S3[4],[QDM_S3[0], QDM_S3[4]]], QDM_S3.subgroups()[1])
```

```
Out[31]: [((2,3), (), (2,3)),
          [((2,3), (), (2,3))],
          [((2,3), (2,3), (2,3))],
          [((2,3), (2,3), (2,3))],
          [((2,3), (), (2,3))],
          ((), (), ()),
          [((), (), ())],
          [((), (2,3), ())],
          [((), (2,3), ())],
          [((), (), ())],
          ((2,3), (2,3), (2,3)),
          [((2,3), (2,3), (2,3))],
          [((2,3), (), (2,3))],
          [((2,3), (), (2,3))],
          [((2,3), (2,3), (2,3))],
          ((), (2,3), ()),
          [((), (2,3), ())],
          [((), (), ())],
          [((), (), ())],
          [((), (2,3), ())]]
```

This implies for the operator $T^{((2,3),e)}$:

Possible initial configuration	*Ground state*
(e,e,e)	$2*(e,e,e)+2*(e,(2,3),e)$
$((2,3),e,(2,3))$	$2*((2,3),e,(2,3))+2*((2,3),(2,3),(2,3))$
$(e,(2,3),e)$	$2*(e,e,e)+2*(e,(2,3),e)$
$((2,3),(2,3),(2,3))$	$2*((2,3),e,(2,3))+2*((2,3),(2,3),(2,3))$

```
In [32]: ground_state_sum([QDM_S3[4],[QDM_S3[1], QDM_S3[2]]], QDM_S3.subgroups()[1])
```

```
Out[32]: [((), (1,2), ()),
          [((), (1,2), ())],
          [((), (1,2,3), ())],
          [((), (1,3,2), ())],
          [((), (1,3), ())],
          ((), (1,2,3), ()),
          [((), (1,2,3), ())],
          [((), (1,2), ())],
          [((), (1,3), ())],
          [((), (1,3,2), ())]]
```

This implies for the operator $T^{((2,3),(1,2))}$:

Possible initial configuration	*Ground state*
$(e,(1,2),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$
$(e,(1,2,3),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$

```
In [33]: ground_state_sum([QDM_S3[4],[QDM_S3[3], QDM_S3[5]]], QDM_S3.subgroups()[1])
```

```
Out[33]: [((), (1,3,2), ()),
          [((), (1,3,2), ())],
          [((), (1,3), ())],
          [((), (1,2), ())],
          [((), (1,2,3), ())],
          ((), (1,3), ()),
          [((), (1,3), ())],
          [((), (1,3,2), ())],
          [((), (1,2,3), ())],
          [((), (1,2), ())]]
```

This implies for the operator $T^{((2,3),(1,2,3))}$:

Possible initial configuration	*Ground state*
$(e,(1,3,2),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$
$(e,(1,3),e)$	$(e,(1,2),e)+(e,(1,2,3),e)+(e,(1,3,2),e)+(e,(1,3),e)$

Therefore, there are 3 unique ground states for all possible configurations of ribbon operators with an excitation at one end and condensate at the other

Bibliography

[1] Jonathan Golan. *The Linear Algebra a Beginning Graduate Student Ought to Know*, The Endomorphism Algebra of a Vector Space:113–131, 2000. URL http://link.springer.com/chapter/10.1007/978-94-007-2636-9_7/fulltext.html.

[2] S. Beigi, P. Shor, and D. Whalen. The quantum double model with boundary: Condensations and symmetries. *Commun. Math. Phys. 306*, pages 663–694, 2011. URL http://arxiv.org/pdf/1006.5479v4.pdf.

[3] Pieter Naaijkens. Anyons in infinite quantum systems: Qft in d=2+1 and the toric code. *Dissertation, Radboud Universiteit Nijmegen*, 2012. URL http://www.math.ru.nl/~landsman/Naaijkens.pdf.

[4] L. Kong. Anyon condensation and tensor categories. *Nuclear Physics*, 886:436–482, 2014. URL http://arxiv.org/pdf/1307.8244v6.pdf.

[5] H. Bombin and M. A. Martin-Delgado. A family of non-abelian kitaev models on a lattice: Topological condensation and confinement. *Phys. Rev. B 78*, 2008. URL http://arxiv.org/pdf/0712.0190v3.pdf.

[6] M. A. Levin and X. G. Wen. String-net condensation: A physical mechanism for topological phases. *Phys. Rev. B 71*, 2005. URL http://arxiv.org/pdf/cond-mat/0404617v2.pdf.

[7] A. Y. Kitaev and L. Kong. Models for gapped boundaries and domain walls. *Commun. Math. Phys. 313*, pages 351–373, 2012. URL http://arxiv.org/pdf/1104.5047v4.pdf.

[8] Yuting Hu, Nathan Geer, and Yong-Shi Wu. Full dyon excitation spectrum in generalized levin-wen models. *eprint arXiv:1502.03433*, 2015. URL http://arxiv.org/pdf/1502.03433v1.pdf.

[9] Alexei Davydov. Modular invariants for group-theoretical modular data 1. *eprint arXiv:0908.1044*, 2009. URL http://arxiv.org/pdf/0908.1044v1.pdf.

[10] Yuting Hu, Stirling Spencer, and Yong-Shi Wu. Ground-state degeneracy in the levin-wen model for topological phases. *Physical Review B Issue 7*, 85, 2012. URL http://arxiv.org/pdf/0908.1044v1.pdf.

[11] Idear. Explicit 2-cocycles of g=z2×z2xz2 over u(1). *mathoverflow*, 2013. URL http://mathoverflow.net/questions/127660/explicit-2-cocycles-of-g-z2%C3%97z2xz2-over-u1.

[12] Beni Yoshida. Gapped boundaries, group cohomology and fault-tolerant logical gates. *eprint arXiv:1509.03626*, 2015. URL http://arxiv.org/pdf/1509.03626v1.pdf.

[13] Beni. Topological phases with generalized global symmetries. *eprint arXiv:1508.03468*, 2016. URL http://arxiv.org/pdf/1508.03468v2.pdf.

[14] Luiz H. Santos and Eduardo Fradkin. Instanton effects in lattice models of bosonic symmetry-protected topological states. *eprint arXiv:1512.06864*, 2016. URL http://arxiv.org/pdf/1512.06864v2.pdf.

[15] Sergey Bravyi and Robert Konig. Classification of topologically protected gates for local stabilizer codes. *eprint arXiv:1206.1609*, 2012. URL http://arxiv.org/pdf/1206.1609v1.pdf.

[16] Jacob Miller and Akimasa Miyake. Hierarchy of universal entanglement in 2d measurement-based quantum computation. *eprint arXiv:1508.02695*, 2016. URL http://arxiv.org/pdf/1508.02695.pdf.

[17] Keisuke Fujii. Quantum computation with topological codes — from qubit to topological fault-tolerance. *eprint arXiv:1504.01444*, 2015. URL http://arxiv.org/pdf/1504.01444v1.pdf.

[18] The Sage Developers. Sage Mathematics Software (Version 6.9.beta7). 2015. URL http://www.sagemath.org.

[19] Yuting Hu, Yidun Wan, and Yong-Shi Wu. Twisted quantum double model of topological phases in two dimensions. *Physical Review B Issue 12*, 87, 2013. URL http://arxiv.org/pdf/1211.3695v2.pdf.

[20] Andrej Mesaros and Ying Ran. Classification of symmetry enriched topological phases with exactly solvable models. *Physical Review B Issue 15*, 87, 2013. URL http://arxiv.org/pdf/1212.0835v3.pdf.

[21] Yidun Wan, Juven C. Wang, and Huan He. Twisted gauge theory model of topological phases in three dimensions. *Physical Review B Issue 4*, 92, 2015. URL http://arxiv.org/pdf/1409.3216v2.pdf.

[22] Jamie Vicary. The topology of quantum algorithms. *LICS 2013, Proceedings of 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 93–102, 2012. URL http://arxiv.org/abs/1209.3917.

[23] William Zeng and Jamie Vicary. Abstract structure of unitary oracles for quantum algorithms. *EPTCS 172, DOI : 10.4204/EPTCS.172.19*, pages 270–284, 2014. URL http://arxiv.org/abs/1406.1278.