Universal Resource States for 1D Measurement-Based Quantum Computation

Jeb SONG

Supervisor: Prof. HO Wen Wei

Assessor: Prof. GONG Jiangbin

UROPS

Abstract

The quantum circuit model is not the only way to understand quantum computing. Measurement-based quantum computation (MBQC) offers an alternative to the circuit-based model. In this report, we discuss tensor networks as a graphical framework for understanding MBQC. Using the rules discussed, we focus on the 1D and 2D cluster states of MBQC and how feed-forward control, depending on the measurement outcomes, can generate some desired logical operations in the logical Hilbert space irrespective of measurement outcome. We then cover the 1D MBQC scheme by Stephen et al. (2022) that extends the 1D cluster state to allow for universal quantum computation by increasing entanglement. We explain how feed-forward control is used in this scheme and how it is universal. Finally, we extend this scheme to qudits and find that for qutrits, computation is always universal, whereas for qubits, it is only universal under certain restrictions.

Contents

1 Introduction **3**
1.1 What is Quantum Computation? . . . 3
1.2 Measurement-Based Quantum Computation . . . 4

2 Tensor Network Notation **5**
2.1 Basics of Tensor Network Notation . . . 5
2.1.1 Bell Pair in Tensor Network Notation . . . 6
2.1.2 Quantum Gates in Tensor Network Notation . . . 7
2.2 Example: Quantum Teleportation . . . 8

3 MBQC **10**
3.1 1D Cluster State . . . 10
3.1.1 Ideal Measurement . . . 11
3.1.2 Feed-Forward Control of 1D Cluster State . . . 11
3.2 2D Cluster State . . . 13
3.2.1 σ^z-measurement . . . 14
3.2.2 σ^x-measurement . . . 14

4 Universal MBQC paper by Stephen et al. (2022) **16**
4.1 Resource State . . . 16
4.2 Ideal Measurement and Dual Unitarity . . . 16
4.3 Feed-Forward Measurement Control . . . 17
4.4 Universality of Logical Unitary . . . 19

5 Generalisation of Stephen et al. (2022) to Qudits **20**
5.1 Generalisation of the Z, X Operators . . . 20
5.1.1 Commutation of X, Z . . . 20
5.2 Generalisation of T_k . . . 21
5.2.1 Correspondence Between the Minimal Lie Algebra and Pauli Matrices . . . 21

6 Algorithm for Finding dim($\mathcal{A}_k$) **23**
6.1 Algorithm Efficiency . . . 24
6.2 Implementation for $d = 2$ . . . 24
6.3 Implementation for $d = 3$ . . . 24

7 Conclusion **26**

8 References **27**

9 Appendix **28**
9.1 Tensor Network Manipulations for Teleportation . . . 28
9.2 Tensor Network Manipulations for 1D Cluster State . . . 29
9.3 Pauli Group and Clifford Group for Qudits . . . 30
9.3.1 Pauli Group and Clifford Group for Qubits . . . 30
9.3.2 Generalisation to Qudits . . . 30
9.4 Lie Algebra for Qubits . . . 31
9.5 Lie Algebra for Qutrits and Qudits . . . 32
9.6 Choice of Qutrit S and Evolution . . . 33

1 Introduction

Since Feynman first proposed the use of quantum computers to simulate quantum systems [1], quantum computers have captured the minds of physicists and sci-fi enthusiasts alike. As the name suggests, quantum computers utilise the power of quantum mechanics. Quantum computers can leverage quantum laws, such as entanglement and superposition, to solve computational problems currently difficult to solve on classical computers.

One of the earliest and most significant demonstrations of quantum computing's potential came in the 1990s, with the development of Peter Shor's quantum algorithm for factoring numbers. This algorithm has the potential to break RSA encryption, which relies on factoring large numbers. Due to this imminent threat, researchers are developing encryption algorithms robust against quantum computers.

The past decade has seen the emergence of a new era of quantum computing, called the Noisy Intermediate-Scale Quantum (NISQ) era. NISQ quantum computers have a limited number of qubits (50-100) and are subject to noise and errors that limit their computational power [1]. However, NISQ computers have already demonstrated their ability to solve problems that are beyond the reach of classical computers.

Looking to the future, the potential impact of quantum computing is enormous. While some remain sceptical about the feasibility of many-qubit quantum computation, many physicists believe that the development of fault-tolerant quantum computers with thousands of qubits is just around the corner. Such quantum computers could have a profound impact on topics such as developing materials to solve the climate crisis, implementing molecular simulations to discovery novel drugs, as well exploring the foundations of quantum mechanics.

1.1 What is Quantum Computation?

Like how classical computers are built upon bits (e.g. transistors which allows or prevents current to pass), quantum computers are based upon qubits (e.g. spin up or spin down state of electron). Classical bits are exclusively "on" or "off" (1 or 0). In contrast, qubits could be in a superposition of two states. Mathematically, qubits can be described by a state-vector $|\psi\rangle$ in the complex Hilbert space, represented as a linear combination of orthonormal vectors $|0\rangle$ and $|1\rangle$ (analogous to the classical states $0, 1$):

$$|\psi\rangle = a|0\rangle + b|1\rangle, \tag{1}$$

where $a, b \in \mathbb{C}$ under the restriction $|a|^2 + |b|^2 = 1$.

Based on these qubits, we can form a particular model of quantum computation, termed the quantum circuit model [1], which is analogous to the circuit model for classical computers. In the quantum circuit model, computation is implemented by initialising a set of qubits in some easy to prepare state, like the $|0\rangle$ state; applying a series of so-called quantum logic gates to the set of qubits (analogous to binary logical gates); then measuring all the qubits in some desired basis (like the σ^z basis, which measures either $|0\rangle$ or $|1\rangle$). We hope that the logic gates available to us form a universal gate set. Such a gate set can generate any possible unitary operation we desire (Miller, 2017).

In general, there are three ingredients to quantum computation: first, the quantum state of some isolated system can be mathematically represented as a vector in the Hilbert Space, and we can reliably prepare qubits in a state represented by some basis within the Hilbert space. Second, we can evolve this system under some unitary U to map our initial configuration of qubits to some output configuration. Third, we can measure the quantum state to obtain frequency statistics in some basis within the Hilbert space [2].

Universal quantum computation is the ability to execute any quantum algorithm on a quantum computer, regardless of its complexity or size. This is analogous to the concept of universal computation in classical computers, where any computation can be performed given enough bits and the ability to manipulate them. Essentially, if we can prepare n-qubits, where n is finite but

arbitrary, and have access to experimental operations which allow us to evolve the system with some n-qubit U, then we can run any quantum algorithm we want.

1.2 Measurement-Based Quantum Computation

Other than the quantum circuit model, are there other models of quantum computation which satisfies these three requirements? Measurement based quantum computing (MBQC) is an equally valid model for quantum computation. In MBQC, the computation is driven by applying single-spin measurements on a carefully-constructed entangled many-body system, known as the resource state. This flips the traditional quantum circuit model on its head, as input states are acted on by unitaries in some logical Hilbert space arising from measurements.

While MBQC does not involve the use of entangling gates beyond the preparation of the resource state, it relies heavily on the physical and analytical properties of the resource state itself to determine which computations can be performed. MBQC is also called the "one-way" quantum computer, since the resource state is consumed during computation via measurement to implement logical operations on the quantum system. On the other hand, the quantum circuit model allows for easy resetting of the circuit, and qubits are not depleted until the final measurements are taken to obtain frequency statistics.

These differences underscore the unique nature of MBQC as a quantum computing paradigm. By utilising on the entangled structure of resource states and employing single-spin measurements instead of entangling gates, novel platforms for quantum computing may arise where MBQC proves more advantageous [3]. MBQC offers an exciting new direction for theoretical insights and experimental design in quantum computing.

2 Tensor Network Notation

Tensor networks are an incredibly useful tool for studying quantum many-body systems in both condensed matter physics and quantum information. They allow us to simplify complex equations into easy-to-understand diagrams, making it easier to grasp the underlying mechanisms of quantum circuits. The language of tensor networks is intuitive, providing both analytical and numerical benefits, as well as a conceptual advantage in helping us understand complex systems.

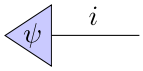
In quantum many-body systems, tensor networks can simplify circuit details, allowing us to perform contractions and simplifications with ease. Moreover, they provide a means for classically simulating certain quantum systems, such as the 1D matrix product state tensor network. Additionally, tensor networks are strong conceptual tools in condensed matter physics. For example, tensor networks can help us classify gapped zero-temperature phases in 1D [4].

In this section, we will introduce the basics of tensor networks and investigate some examples. In Section 2.1, we will cover the basic representations of tensors and tensor contractions. In Subsection 2.1.1, we will describe how to represent the Bell pair, an important entangled state in the context of quantum communication. In Subsection 2.1.2, we will prove that the quantum teleportation protocol works using the tensor network manipulations we discussed. For our purposes, tensor networks give us a way to manipulate and understand MBQC and easily visualize the effects of measurements.

2.1 Basics of Tensor Network Notation

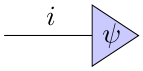
Tensors are mathematical objects that generalize vectors and matrices to arbitrary dimensions. A tensor of rank n in $d_1 \times \cdots \times d_n$ dimensions has the expression $T_{i_1,\ldots,i_n} \in \mathbb{C}^{d_1 \times \cdots \times d_n}$. For example, a vector is a rank one tensor, and a matrix is a rank two tensor.

In tensor network notation, we represent tensors as shapes with legs sticking out, where each leg represents an index of the tensor. For example, the tensor network representation of a single qubit statevector $|\psi\rangle$ is:



(2)

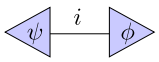
where $i \in \{0, 1\}$. For choice of the index i, the diagram will return the component $\langle i|\psi\rangle$. The complex conjugate, $\langle\psi|$, of this statevector has the representation:



(3)

which is simply a change in the direction the leg is sticking out of the triangle. Similarly, $i \in \{0, 1\}$ such that for some choice of the index i, the diagram will return the component $\langle\psi|i\rangle$. Thus, the direction the legs stick in tensor networks inform us whether the vector is a bra or ket.[1]

The contraction of the indices for some $|\psi\rangle$ and $\langle\phi|$ has the representation:



(4)

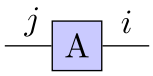
and since there are no legs sticking out of the resulting shape, we know that it represents a scalar. The contraction of the index i gives:

$$\sum_{i=0}^{1} \langle\phi_i|\psi_i\rangle\,, \tag{5}$$

which is precisely the inner product $\langle\phi|\psi\rangle$.

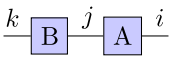
[1] In conventional tensor network notation, the ket vector has legs pointing to the left, and vice versa. We have reversed the directions for all tensors to maintain consistency with the rest of the report.

The matrix A_{ij} will have the representation:



(6)

where the number of legs sticking out determine the tensor rank. On the other hand, the contraction of two matrices A, B by one index has the representation:

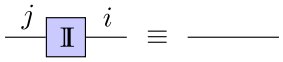


(7)

The resultant shape has two legs sticking out on the remaining indices, meaning that it's a rank two tensor. This operation is equivalent to matrix multiplication, as the contraction of the index j is equivalent to summation over the index j in the resulting tensor:

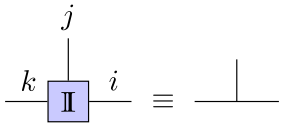
$$\sum_j A_{ij} B_{jk} = (AB)_{ik}. \tag{8}$$

The identity is easily represented in tensor network notation:



(9)

where the above action is equivalent to the Kronecker delta δ_{ij}, as we force the index to be the same across the leg indices. Similarly, the rank 3 identity has the representation:



(10)

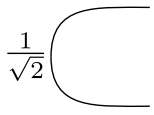
where the above action is similar to Kronecker delta δ_{ijk}. From the above examples, we can extend the representations of the identity and Kronecker delta to arbitrary rank.[2]

2.1.1 Bell Pair in Tensor Network Notation

The Bell pair, which is the simplest example of entanglement between two qubits, plays a crucial role in quantum communication protocols such as quantum teleportation. The Bell pair $|\beta_{00}\rangle$ is defined as follows:

$$|\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}. \tag{11}$$

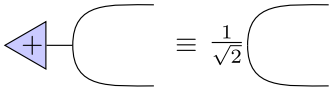
Using tensor network notation, the Bell pair is represented as:



(12)

and by substituting the identity or Kronecker delta δ_{ij} to the left side of the diagram, we can obtain the components equivalent to the Bell pair. Specifically, feeding in 0 in the top index means we feed in 0 in the bottom index, and vice versa. However, since $\delta_{ij} = 1$ for $i = j$, we need to multiply the whole diagram by $\frac{1}{\sqrt{2}}$ to obtain the values equivalent to the Bell pair.

An equivalent representation of the Bell Pair is:



(13)

[2] See https://arxiv.org/abs/1912.10049 for a pedagogical exposition of tensor networks

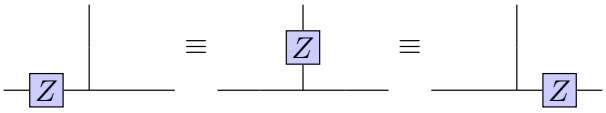
where the tensor denoted with the + is the statevector $|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. This identity can be checked for by getting the inner product of Equation 13 with $|00\rangle, |11\rangle$, which are precisely the components for $\langle 00|\beta_{00}\rangle$ and $\langle 11|\beta_{00}\rangle$. Since the constants in the diagram are inconsequential to the resulting mechanism of the tensor network, we may ignore them. Thus, within the tensor network notation, the above two diagrams are equivalent.

Importantly, the $|+\rangle$ tensor can be freely attached to any leg, like Equation 13. We can freely remove and add the $|+\rangle$ tensors to the legs. This property will prove very useful in our understanding of MBQC.

2.1.2 Quantum Gates in Tensor Network Notation

Single qubit gates from the quantum circuit model can be represented as rank two tensors in tensor network notation (see Equation 6). There are some additional identities which these gates obey, as described below.

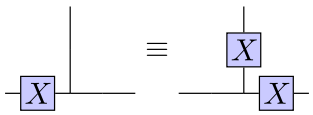
For Z-gate, since is a diagonal matrix, we know that it obeys:



(14)

which can be checked, as Z is non-zero only when the row and column indices are the same. Just like the Kronecker delta, it "force" the connected legs to have the same indices. In general, the above identity holds for all diagonal matrices which are rank 2 or more.

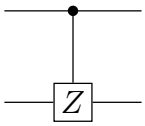
Additionally, for the X-gate, it obeys the identity below:



(15)

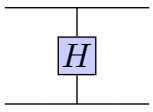
which is easily checked by plugging in the indices.

For the control-Z gate, the quantum circuit notation is:



(16)

and the equivalent expression in tensor network notation, up to a multiplicative constant of $\frac{1}{\sqrt{2}}$, is:



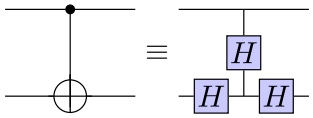
(17)

which is possible because the Hadamard gate has the matrix representation:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{18}$$

where the Hadamard gate returns a negative value when both the first and second index are 1, similar to the action of the control-Z gate.

Since $CX_{1,2} = H_2 CZ_{1,2} H_2$, we know that the Control-X gate has the representation in tensor network notation:



(19)

and these representations of the CZ and CX gates will be useful in manipulations of tensor networks.

Using the above rules and representations of the gates, we can now consider the quantum teleportation protocol in tensor network notation, which will demonstrate the power of tensor network notation for simplifying circuits.

2.2 Example: Quantum Teleportation

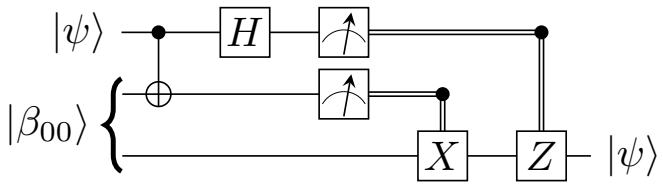


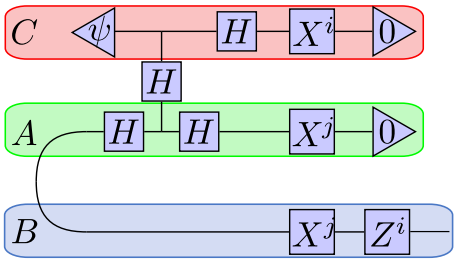
Figure 1: The quantum teleportation protocol in quantum circuit notation. The first qubit represents Charlie's qubit with statevector $|\psi\rangle$. $|\beta_{00}\rangle$ is the Bell pair, where the second qubit belongs to Alice, the third qubit to Bob.

Imagine that Charlie, Alice, and Bob each had a qubit on Earth. Suppose that Charlie and Alice stayed on earth, and that Bob was sent on a spaceship far away. Suppose Charlie has a qubit in some unknown state $|\psi\rangle$. He wants to send this qubit to Bob, who is far away on a spaceship. However, Bob does not have the gates required to generate $|\psi\rangle$, so Charlie and Alice must find another way to send the qubit to Bob. What could Charlie and Alice do?

The preparation procedure to allow this is called the quantum teleportation protocol. To do this, Alice and Bob began by each obtaining a qubit from a Bell pair. Thus, their qubits will be entangled. When Bob goes on his spaceship, Alice and Charlie performs a set of gates on their qubits. Next, Charlie and Alice each measure their qubit in the σ^z basis. Depending on the outcome of their measurement, they get one of four possible classical results: 00, 01, 10, or 11. Finally, Alice and Charlie sends their two classical measurement outcomes to Bob, who then performs a set of gates on his qubit depending on the results. These gates "fix" the state of Bob's qubit so that it becomes $|\psi\rangle$.

In summary, the quantum teleportation protocol allows Charlie to send his qubit $|\psi\rangle$ to Bob by using shared entanglement and classical communication. Charlie is able to "teleport" the quantum state of his qubit to Bob's qubit. As Nielsen and Chuang note, utilising entanglement in quantum systems opens up "a new world of possibilities unimaginable with classical information" (p. 26). A general proof that the protocol works is somewhat tedious. However, in tensor network notation, the proof is simple.

The equivalent teleportation circuit in tensor network notation is:



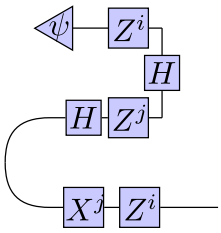
(20)

where the labels C, A, B represent Charlie, Alice, and Bob's qubit respectively.

In tensor network notation, a measurement is represented as a projection onto some basis (such as the computational basis σ^z). Specifically, when measuring the first qubit in the state $|0\rangle$, we

assign $i = 0$, while $i = 1$ when measuring the first qubit in the state $|1\rangle$. The same convention is used for index j from the second qubit. Informed by the measurement outcomes i, j, Bob performs a correction operation $X^j Z^i$, which returns the state $|\psi\rangle$ he is interested in. The operators X^i and X^j are determined by the measurement outcome, and they account for all possible outcomes in the protocol.

Under tensor network manipulations, the tensor network from Equation 20 becomes (see Appendix 9.1 for explicit tensor network manipulation):

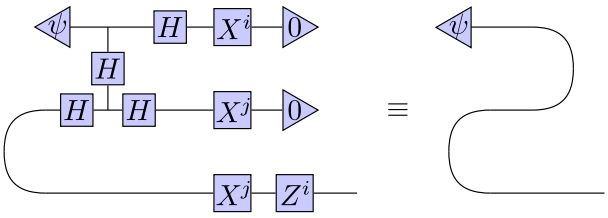


(21)

We know that:

$$Z^i X^j H Z^j H Z^i = Z^i X^j X^j H H Z^i = Z^i Z^i = \mathbb{1}. \tag{22}$$

Thus, we can establish the equivalence between the two diagrams:



(23)

These manipulations graphically demonstrates how the whole teleportation circuit is equivalent to moving the statevector $|\psi\rangle$ from Charlie to Bob. Using a few tensor network manipulations, we have proved that the quantum teleportation protocol works.

3 MBQC

The fundamental ingredients of MBQC involve the resource state, a many-body entangled system of qubits; measurements which act as logical unitaries on the circuit; and feed-forward control of measurements. Such control ensures that the desired logical unitary is implemented independent of measurement outcome.

In this section, we highlight two basis schemes for MBQC. Standard texts view MBQC through the lens of stabilizer formalism or graph theory. This section will cover MBQC in terms of tensor networks, which is easier to appreciate. In Section 3.1, we consider the 1D cluster state. We will show how the 1D cluster state can simulate any unitary acting on one qubit, considering the ingredients of MBQC. In Section 3.2, we consider the 2D cluster state. Based on our investigation of the 1D cluster state and some new operations, we will demonstrate that the 2D cluster state allows for universal quantum computation.

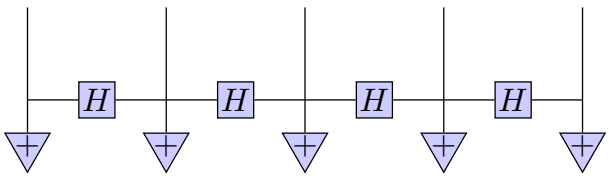
3.1 1D Cluster State

As described in [2], the 1D cluster state $|\psi_{1Dn}\rangle$ of n qubits is formed by initialising n qubits in some 'scratch' state $|0\rangle^{\otimes n}$, then applying Hadamard gates to each qubit, given as $H^{\otimes n}$. This will give us the state $|+\rangle^{\otimes n}$. Then, we implement CZ gates on neighbouring qubits.

Thus, the cluster state is of the form:

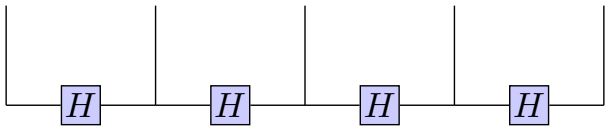
$$|\psi_{1Dn}\rangle = \left(\prod_{i=1}^{n-1} CZ_{i,i+1}\right)|+\rangle^{\otimes n}. \tag{24}$$

For $n = 5$, the tensor network representation of the 1-D cluster state $|\psi_{1D5}\rangle$ is:



(25)

From Equation 13, we can remove the $|+\rangle$ tensors to get the equivalent representation:



(26)

To simulate any unitary on two qubits, we will need to prepare $n = 5$ qubits. This is based on the protocol from [5]. For the protocol, the first qubit (from the left) is measured in the σ^z-basis with measurement outcomes $|0\rangle, |1\rangle$. The second, third, and fourth qubits are measured in the XY-plane. Explicitly, these qubits are measured in the basis $\{|0^\theta\rangle, |1^\theta\rangle\}$ determined by some angle θ, where:

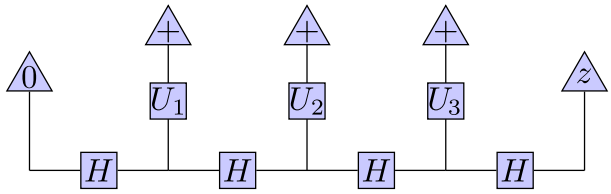
$$|0^\theta\rangle = e^{i(\theta/2)Z}|+\rangle, |1^\theta\rangle = e^{i(\theta/2)Z}|-\rangle, \tag{27}$$

and the fifth qubit is measured in the σ^z-basis with an unknown outcome, $z \in \{0, 1\}$.

In quantum mechanics, we know that our measurements are probabilistic. If in general we measure some quantum state in $|0\rangle$ with probability p, we know that we will measure $|1\rangle$ with probability $1 - p$. Likewise for the basis $\{|0^\theta\rangle, |1^\theta\rangle\}$. In the subsequent section, we consider the ideal case, where the measurement outcome for the first qubit is $|0\rangle$, and the measurement outcome for the subsequent three qubits is $|0^{\theta_j}\rangle$. This choice helps simplify our tensor network manipulations.

3.1.1 Ideal Measurement

The tensor network diagram for the whole protocol is:



(28)

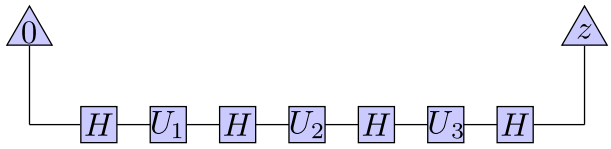
where $U_j = e^{i(\theta_j/2)Z}$. Explicitly, U_j is some rotation along the Z-axis of the Bloch sphere with the matrix representation:

$$U_j = \begin{bmatrix} e^{-i\theta_j/2} & 0 \\ 0 & e^{i\theta_j/2} \end{bmatrix}. \tag{29}$$

The above diagram from Equation 28 is equivalent to the inner product:

$$\langle 00^{\theta_1}0^{\theta_2}0^{\theta_3}z|\psi_{1D5}\rangle\,. \tag{30}$$

Simplifying the diagram by removing the $|+\rangle$ tensors (using Equation 13) and moving down the diagonal U_j, we have the diagram:



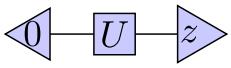
(31)

We know that:

$$He^{i(\theta_3/2)Z}He^{i(\theta_2/2)Z}He^{i(\theta_1/2)Z}H = e^{i(\theta_3/2)X}e^{i(\theta_2/2)Z}e^{i(\theta_1/2)X}, \tag{32}$$

which is precisely the Euler angle decomposition of single qubit unitaries, which implies that we can simulate any one-qubit unitary U in the ideal case where we measure $|0\rangle, |0^\theta\rangle$ for first qubit and the subsequent three qubits respectively.

Thus, the above diagram simplifies to:



(33)

which is the inner product:

$$\langle z|U|0\rangle\,, \tag{34}$$

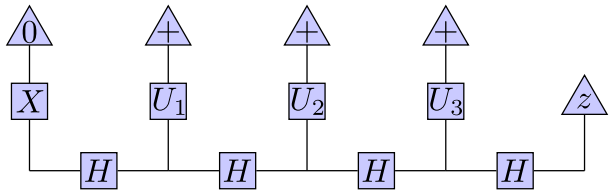
where $U = e^{i(\theta_3/2)X}e^{i(\theta_2/2)Z}e^{i(\theta_1/2)X}$. From these tensor network manipulations, we conclude that $\langle 00^{\theta_1}0^{\theta_2}0^{\theta_3}z|\psi_{1D5}\rangle$ is equivalent to $\langle z|U|0\rangle$.

We can also establish a relation between the physical space (number of physical qubits) of the circuit from Equation 28 with the logical time (number of logical operations) of the circuit from Equation 33. Measuring the second, third, and fourth physical qubits corresponds to logical rotations of the logical qubit.

3.1.2 Feed-Forward Control of 1D Cluster State

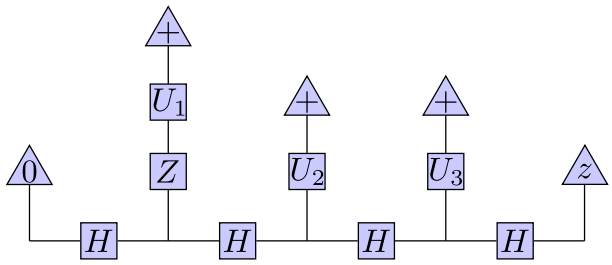
We now consider the general case where the measurement outcome is not necessarily $|0\rangle$ and $|0^\theta\rangle$ due to the random measurement outcomes. We will show that even in the case of different measurement outcomes, we can still deterministically change subsequent measurements to generate the logical U we desire.

As an example, if we measure $|1\rangle$ on the first qubit, our diagram will look like:



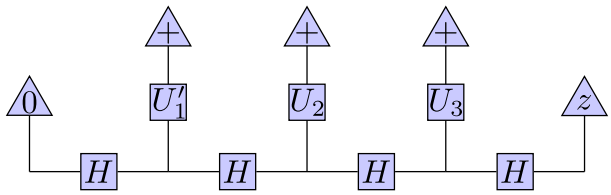
(35)

moving the X tensor over the H operator, we will get the diagram:



(36)

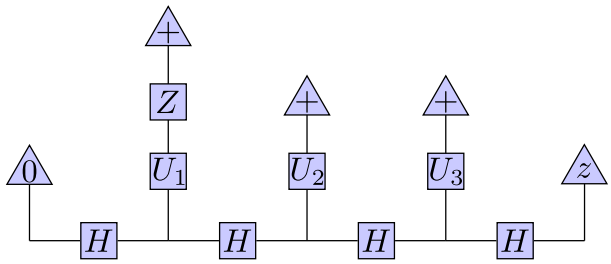
which we represent as:



(37)

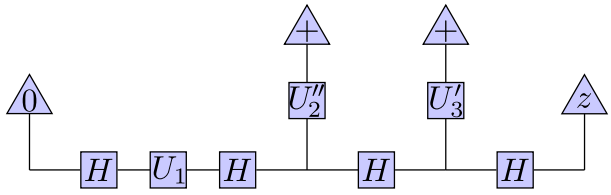
where $U_1' = e^{i(\theta_1/2+\pi/2)Z}$. Thus, the measurement outcome of the first qubit will change the logical unitary implemented by the measurement on the second qubit. If we desire to implement some rotation θ on the second qubit, measuring $|1\rangle$ for the first qubit means that we should instead measure the rotation $\theta - \pi$ for the second qubit. Under such measurement controls, the diagram from Equation 35 is equivalent to the ideal case in Equation 28.

As another example, we consider the case where we measure $|1^{\theta_1}\rangle$ for the second qubit after measuring the first qubit as $|0\rangle$. In this case, the diagram will look like:



(38)

since we measure $|1^{\theta_1}\rangle = e^{i(\theta_1/2)Z}|-\rangle = e^{i(\theta_1/2)Z}Z|+\rangle = U_1 Z|+\rangle$ on the second qubit. Implementing tensor network manipulations we have described (see Appendix 9.2), the above diagram simplifies to:



(39)

where $U_2'' = e^{-i(\theta_2/2)Z}$ and $U_3' = e^{i(\theta_3/2+\pi/2)Z}$.

Thus, If we desire to implement rotations ξ, η on the second and third qubit respectively, measuring $|1^{\theta_1}\rangle$ for the second qubit means that we should instead measure the rotations $-\xi$ and $\eta - \pi$ for the third and fourth qubits respectively. Under such measurement controls, the diagram from Equation 39 is equivalent to the ideal case in Equation 28.

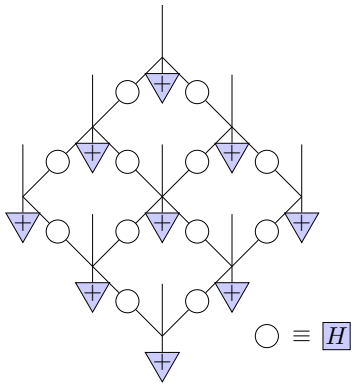
The logic for the feed-forward controls for the third and fourth qubit is the same. Thus, we have shown that no matter the random outcomes of measurement, we can reliably implement the desired logical U from the ideal case by controlling subsequent measurements (either by taking the negative angle of rotation, or measuring with phase $-\pi$) based on previous measurements. Experimentally, it is easy to keep track of the changes required for subsequent measurements. Even though the measurements are random, we can reliably generate our desired outcome within MBQC.

3.2 2D Cluster State

Though the 1D cluster state allows us to generator arbitrary logical rotations around one logical qubit, this won't allow us to perform universal quantum computation, as we cannot scale up the quantity of logical qubits.

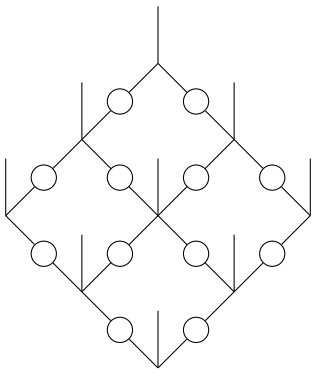
The 2D cluster state is the generalisation of the 1D cluster state to 2D. Implementing the 2D cluster state requires a physical 2D array, with entanglement between qubits. Theoretically, the 2D cluster state can implement any multi-qubit unitary on any number of qubits we want desire. Thus, we can perform universal quantum computation on the 2D cluster state.

Just like in the 1D case, we prepare an array of $n \times m$ qubits of n qubits length and m qubits width in the state $|+\rangle^{\otimes n \times m}$. We then implement the CZ gate between horizontally or vertically neighbouring qubits [2]. In tensor network notation, a 3×3 2D cluster state looks like:



(40)

and using the rule for removing the $|+\rangle$ tensor from Equation 13, we know that this is equivalent to:

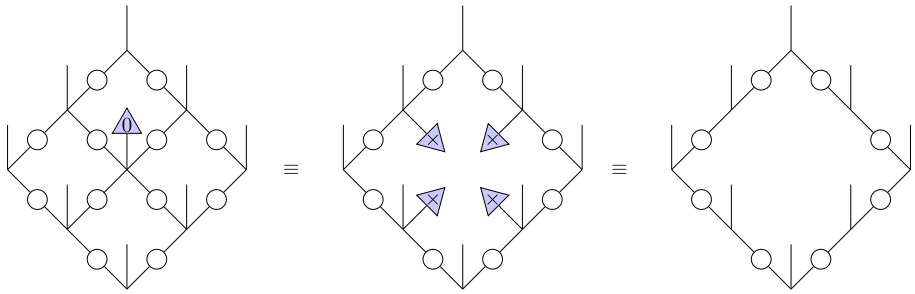


(41)

In addition to the measurements available to us in the 1D cluster state, there are two important operations for the 2D Cluster state described below.

3.2.1 σ^z-measurement

We again consider the 3×3 2D cluster state. If we measure the qubit in the middle in the σ^z basis and get the $|0\rangle$. The tensor network diagram for this operation looks like:

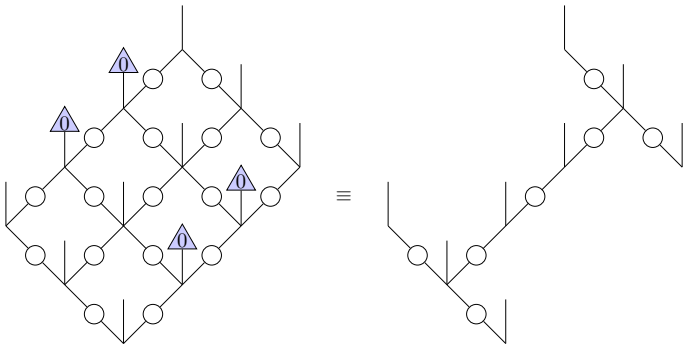


(42)

Thus, measuring in the σ^z basis is equivalent to removing the qubit from the cluster state. σ^z allows us to carve out the geometries of the cluster state. If we instead measure $|1\rangle$, we will have to correct for the Z tensor in the qubits adjacent to the measured qubit.

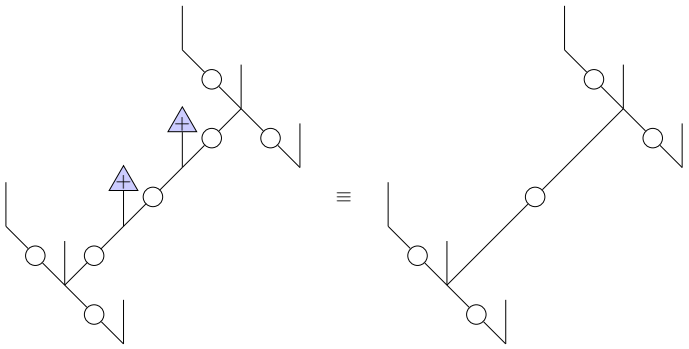
3.2.2 σ^x-measurement

Consider the 3×4 2D cluster state. We measure certain qubits in the σ^z basis to carve out the array:



(43)

in which all measurements are $|0\rangle$ (though we could get the same outcome by implementing feed-foward measurement control). Suppose we measure both qubits in the middle chain in the σ^x basis and obtain $|+\rangle$. Then, the tensor network simplifies to:



(44)

Therefore, if we measure these two qubits in the middle chain in σ^x basis, it is equivalent to implementing a CZ gate between the qubits that are adjacent to the connection from each chain. If we measure the qubits in the $|-\rangle$ state, we must correct for the Z operator on the adjacent qubits. This operation is useful because it allows us to implement a logical CZ gate between logical qubits.

Thus, by incorporating σ^z and σ^x operations, we can carve out 1D chains of qubits with CZ entanglement between them. Using this, we can implement any single qubit logical operation U on the 1D chain (as detailed in Section 3.1) and CZ gates between logical qubits. By scaling the number of qubits along the length and width of the 2D cluster state and carving out the array's geometry based on the multi-qubit unitary's decomposition, we can implement any quantum computation within its logical Hilbert space.[3] In other words, we can implement any multi-qubit unitary operation on any number of qubits by using the 2D cluster state. Therefore, the 2D cluster state is considered a universal resource state for quantum computation.

We can also relate its physical dimensions to logical space and time dimension to understand how the 2D cluster state's role enables universal quantum computation. Specifically, we can choose one physical space dimension to represent logical space and the other to represent logical time. By increasing the number of physical qubits along the physical dimension corresponding to logical space, we can simulate any number of logical qubits by carving out 1D lines along that dimension (with connections between lines to implement entanglement). Furthermore, by evolving the states of these logical qubits through single-spin measurements along the dimension that represents logical time, we can implement any desired unitary operation by scaling up the corresponding physical dimension. This intuitive representation of the 2D cluster state allows us to see why it is a universal resource state for quantum computation. Through tuning the logical space and time dimensions, we can perform universal quantum computation with a high degree of control and flexibility.

In MBQC, discovering such universal resource states can be challenging, as it often relies on specific properties of the state in question. In the subsequent sections, we will explore a particular 1-D MBQC scheme that serves as a universal resource state under certain conditions. We will also attempt to generalize this scheme from the qubit case to the qudit case.

[3] See the image from https://commons.wikimedia.org/wiki/File:Notation-as-it-relates-to-a-one-way-quantum-computation-3-copyright-2001-by-the-APS.png for a pictorial representation of this correspondence

4 Universal MBQC paper by Stephen et al. (2022)

MBQC relies on the entanglement between different regions of a quantum state to perform operations. In the case of the 1D cluster state, which is the most commonly used resource state for MBQC, the entanglement is relatively weak, consisting of only one layer of Controlled-Z gates. As a result, only one qubit of information can be transmitted through physical qubits. If we controllably increase entanglement, we might be able to do more operations and implement more useful logical computations in the 1D. This is the central motivation behind Stephen et al.'s research.

In a recent paper by Stephen et al. (2022) [6], a new universal resource state was introduced for 1D MBQC. The 1D scheme for MBQC provides an universal resource state, depending on certain parameters, that supports multiple logical qubits. By choice of the 1D setup, this scheme offers practical advantages for universal MBQC by eliminating the need for 2D geometries, which simplifies the requirements for potential implementation in experimental platforms. Moreover, the scheme only requires simple and uniform controls, making it particularly relevant to trapped ions, which have high gate and measurement fidelities, long coherence times, and are typically restricted to 1D geometry.

In Section 4.1, we will discuss the resource state used in the 1D MBQC scheme proposed by [6]. In Section 4.2, we will explore the effect of measurement in the ideal case and demonstrate the dual-unitary property of the circuit. In Section 4.3, we will delve into feedforward measurement controls based on different measurement outcomes, and how this affects the logical outcome of the inner product. Finally, in Section Finally, in Section 9.4, we will discuss the universality of the scheme and the underlying mathematical principles that make it possible.

4.1 Resource State

In the proposed scheme from [6], we start by preparing N qubits, where the value of N plays a crucial role in the logical computation we will perform. The qubits are subjected to a Floquet evolution denoted by U_f from the kicked Ising model. The operator U_f is physically equivalent to a unitary operator composed of Clifford operators, given by:

$$T_N = \prod_{i=1}^{N} H_i S_i \prod_{i=1}^{N} CZ_{i,i+1}, \tag{45}$$

where H is the Hadamard gate, $S = \sqrt{Z}$ is the phase gate, and $CZ_{i,i+1}$ is the controlled-Z gate between qubits i and $i+1$.

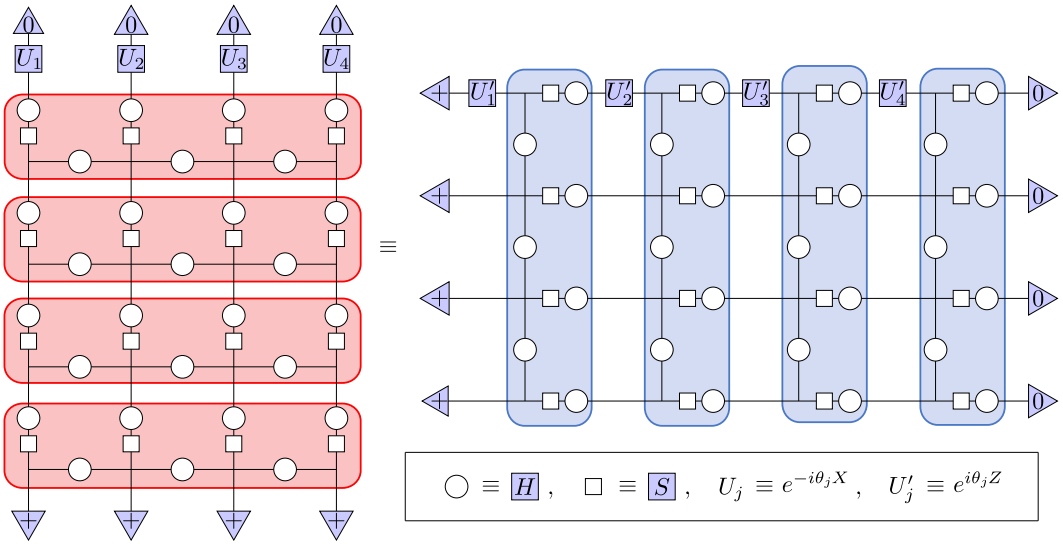
We define the resource state $|\psi_N\rangle$ as:

$$|\psi_N\rangle = T_N^k |+\rangle^{\otimes N}, \tag{46}$$

where we apply the unitary operator T_N on the state $|+\rangle^{\otimes N}$ k times. The value of k also uniquely determines the type of logical computation that can be performed using the resource state.

4.2 Ideal Measurement and Dual Unitarity

To implement MBQC on the resource state, we measure the N qubits in the YZ-plane (left to right for diagram in Equation 47). Explicitly, the qubits are measured in the basis $\{|0^\theta\rangle, |1^\theta\rangle\}$ determined by some angle θ. For shorthand, we write $s_i \in \{0,1\}$, s.t. $|s_i^{\theta_i}\rangle = e^{-i\theta_i X}|s_i\rangle$ for measurement of the i-th qubit. Similar to Subsection 3.1.1 for the 1D cluster state, we first consider the case where the measurement outcome for all qubits is $s_i = 0$. The tensor network diagram of the circuit will then look like the left diagram below from Equation 47. Using tensor network manipulations ($S = \sqrt{Z}$ and U_j' are both diagonal matrices), we can establish logical equivalence with the right diagram below from Equation 47. For $N = 4, k = 4$, the tensor network diagram for the 1D MBQC scheme is:



(47)

The inner product represented by the left diagram is:

$$\langle 0^{\theta_1} 0^{\theta_2} 0^{\theta_3} 0^{\theta_4} | \psi_N \rangle = \langle 0^{\theta_1} 0^{\theta_2} 0^{\theta_3} 0^{\theta_4} | T_N^k | + \rangle^{\otimes N} = \langle 0^{\theta_1} 0^{\theta_2} 0^{\theta_3} 0^{\theta_4} | T_4^4 | + \rangle^{\otimes 4}, \tag{48}$$

while the inner product represented by the right diagram is:

$$\langle 0 |^{\otimes k} U(\theta_N) \cdots U(\theta_1) | + \rangle^{\otimes k} = \langle 0 |^{\otimes 4} U(\theta_4) \cdots U(\theta_1) | + \rangle^{\otimes 4}, \tag{49}$$

where we define the unitary operator $U(\theta_j) = T_k e^{i\theta Z_1}$.

Our choice of N and k has a crucial impact on the computation performed in the MBQC protocol. In the physical setup, N corresponds to the number of physical qubits, which determines the physical space dimension. Meanwhile, k corresponds to the number of times the gate T_N is applied, which determines the physical time dimension. However, in the logical computation, the roles of N and k are reversed: N corresponds to the number of times the gate T_k is applied, while k corresponds to the number of logical qubits.

We also know that the T_N (and correspondingly the T_k) gates in the tensor network diagrams from Equation 47 can be read either from bottom to top (T_N in the physical time direction) or from right to left (T_k in the logical time direction) and is unitary in both cases. This interchange is known as "dual-unitarity". Dual-unitary proves to be a powerful tool in the analysis and design of MBQC protocols.

4.3 Feed-Forward Measurement Control

Suppose at some index $j = l$, we measure $s_j = 1$. Then, the inner products become:

$$\langle 0^{\theta_1}, \ldots, 1^{\theta_j}, \ldots, 0^{\theta_N} | \psi_N \rangle \equiv \langle 0 |^{\otimes k} U(\theta_N) \cdots U(\theta_l) Z_l \cdot \cdots \cdot U(\theta_1) | + \rangle^{\otimes k}. \tag{50}$$

We aim to move the Z_1 operator acting on the first logical qubit to the left side of all the operators. First, we know that $U_l Z_1 = Z_1 U_l$. Then, we also know that:

$$T_k Z_1 = (T_k Z_1 T_k^\dagger) T_k, \tag{51}$$

and since elements of the Clifford group $\mathcal{C}_k$ normalises elements of the Pauli group, we know that $(T_k Z_1 T_k^\dagger)$ must be a Pauli operator (see Appendix 9.3.1 for discussion). The choice of T_N, T_k to be Clifford as important for our scheme. It allows us to reliably determine how different measurements which affect subsequent measurements.

In Subsection 3.1.2 where we covered the feed-forward measurement controls for the 1D cluster state, we saw how the Z, X operators predictably changed the subsequent measurements. Since moving the Z_1 (or any Pauli operator) through the U_j, T_k operators still makes the resulting operator Pauli, different measurement outcomes will have easy to control effects for subsequent measurements (for example, taking the negative of the measurement angle). Without this Clifford property of T_k, we will not have such fine control over different measurement outcomes.

Importantly, the Pauli operators generated from conjugation are given by the local rules:

$$\begin{aligned} T_k Z_i T_k^\dagger &= X_i, \\ T_k X_i T_k^\dagger &= \begin{cases} Y_1 X_2 & i = 1, \\ X_{i-1} Y_i X_{i+1} & 1 < i < k, \\ X_{k-1} Y_k & i = k, \end{cases} \end{aligned} \tag{52}$$

up to some global phase. Using these rules, we can compute arbitrary evolution of the Pauli matrix under conjugation by T_k. After conjugating Z_1 by T_k to some power, these rules help us determine whether the operator X_1 acts on the first qubit or not. For example, $T_k Z_1 T_k^\dagger = X_1$. Conjugating by T_k again, we have up to some global phase:[4]

$$T_k^2 Z_1 \left(T_k^2\right)^\dagger = T_k X_1 T_k^\dagger = Y_1 X_2 = Z_1 X_1 X_2. \tag{53}$$

Finally, the conjugation of Z_1 by T_k^3 gives:

$$\begin{aligned} T_k^3 Z_1 \left(T_k^3\right)^\dagger &= T_k Z_1 X_1 X_2 T_k^\dagger \\ &= \left(T_k Z_1 T_k^\dagger\right)\left(T_k X_1 T_k^\dagger\right)\left(T_k X_2 T_k^\dagger\right) \\ &= (X_1)(Y_1 X_2)(X_1 Y_2 X_3) \\ &= (X_1)(Z_1 X_2)(X_1 Z_2 X_2 X_3) \\ &= Z_1 Z_2 X_3. \end{aligned} \tag{54}$$

Informed by the above examples, we can numerically calculate whether the X_1 operator acts on the first qubit or not for some arbitrary $T_k^a Z_1 \left(T_k^a\right)^\dagger$ by keeping track of the Pauli operators (as was the case for the first and second powers of T_k). Then, we know that:

$$U(\theta_j) T_k^{j-l} Z_1 \left(T_k^{j-l}\right)^\dagger = T_k^{j-l+1} Z_1 \left(T_k^{j-l+1}\right)^\dagger U(\pm\theta_j), \tag{55}$$

where $U(-\theta_j)$ when $T_k^{j-l} Z_1 \left(T_k^{j-l}\right)^\dagger$ has the X_1 operator acting on the first qubit, while $U(\theta_j)$ for the converse. This results in the expression:

$$\begin{aligned} &\langle 0|^{\otimes k} U(\theta_N) \cdots U(\theta_l) Z_1 \cdots U(\theta_1) |+\rangle^{\otimes k} \\ &= \langle 0|^{\otimes k} T_k^{N-l+1} Z_1 \left(T_k^\dagger\right)^{N-l+1} \tilde{U}(\theta_N) \cdots \tilde{U}(\theta_l) U(\theta_{l-1}) \cdots U(\theta_1) |+\rangle^{\otimes k}, \end{aligned} \tag{56}$$

where we have $\tilde{U}(\theta_j) = U(\pm\theta_j)$. The sign depends on whether the Pauli operator we move over $U(\theta_j)$ has the X_1 operator as described.

As [6] demonstrates, all possible measurement outcomes of the physical quits will turn $\langle 0|^{\otimes k}$ to all possible eigenvectors of the σ^z-basis. Thus, we can realise:

$$\langle z | U(\theta_N) \cdots U(\theta_1) |+\rangle^{\otimes k}, \tag{57}$$

where $|z\rangle$ in any eigenvector of the k-qubit σ^z-basis. Interestingly, physical measurements within the scheme are able to simultaneously evolve the logical qubits and help us probe the frequency statistics of the logical qubits.

[4] We write $=$ instead of here $\equiv$, take note that we do not care about the global phases as this won't affect our feed-forward mechanism.

4.4 Universality of Logical Unitary

Since the Clifford group is a finite group, we know that the Clifford T_k must have some finite order :

$$p_k = |T_k| \implies T_k^{p_k} = \mathbb{I}, \tag{58}$$

for which we set $N = p_k$ or some multiple of p_k (as we may need more periods to implement the unitaries we desire).[5] The logical unitary operator we implement becomes:

$$\prod_{j=1}^{p_k} T_k U_{p_k-j+1} = \prod_{j=1}^{p_k} T_k e^{i\theta_{p_k-j+1}Z_1} \propto \prod_{j=1}^{p_k} e^{i\theta_{p_k-j+1}O_k(p_k-j+1)}, \tag{59}$$

where $O_k(p_k - j + 1) = (T_k^\dagger)^{p_k-j} Z_1 T_k^{p_k-j}$, which is governed by rules similar to Section 4.3. We denote the set $\mathcal{O}_k$ as containing elements $O_k(1), \ldots, O_k(p_k)$, which belong to the Pauli group and are distinct from one another (since p_k is the order of T_k).

As [6] describes, for a small angle $d\theta$ and certain operators $A, B \in \mathcal{O}_k$, it is possible to approximate the composite unitary:

$$e^{id\theta A} e^{id\theta B} e^{-id\theta A} e^{-id\theta B} \approx e^{i(d\theta)^2 i[A,B]}, \tag{60}$$

where $[A, B] = AB - BA$ denotes the commutator of A and B (see Appendix 9.4). We can also view the elements $O_k(j) \in \mathcal{O}_k$ as elements of a Lie algebra $\mathcal{A}_k$ where $\mathcal{A}_k \subseteq \mathfrak{su}(2^k)$. The Lie algebra $\mathcal{A}_k$ is obtained by closing the set $\mathcal{O}_k$ under commutation. We would like to find the dimension of this Lie algebra, as we can implement any unitary in $\mathrm{SU}(2^k)$ of the form $R = e^{iA}$, where $A \in \mathcal{A}_k$. If $\mathcal{A}_k = \mathfrak{su}(2^k)$, then we can in principle generate any element in $\mathrm{SU}(2^k)$. If this is the case, our resource state could run any quantum algorithm requiring k qubits. As [6] shows, this is precisely the case when $k = 3, 6$. Moreover, the researchers proved that for any given k, there exists approximately $m \approx \frac{3k}{4}$ logical qubits where arbitrary rotations are achievable. In other words, while rotations may not necessarily universal for arbitrary k values, we can examine the embedded logical qubits for which we can attain universal control.

See Appendix 9.4 for a brief discussion on Lie algebras for qubits.

[5] For our purposes we consider $N = p_k$, though the calculations are the same for $N = a \cdot p_k$ for some $a \in \mathbb{N}$

5 Generalisation of Stephen et al. (2022) to Qudits

In this report, we will build upon the work of [6] by exploring the qudit generalisation of T_k. A qudit is a d-dimensional quantum system that can be represented by d basis states. As an example, a qubit is a two-dimensional qudit can be represented by 2 basis states, for example $|0\rangle$ and $|1\rangle$.

In Section 5.1, we will introduce the generalisation of the Pauli Z and X operators to arbitrary dimensions. We will also investigate the commutation relations between qudit Pauli operators in Subsection 5.1.1. In Section 5.2, we will extend the results of the paper to qudits. Specifically, we will describe a relationship between the Lie algebras for qudits and the generalised Pauli matrices. This relationship will allow us to easily generalise the results obtained in the paper for qubits to higher dimensional qudits.

5.1 Generalisation of the Z, X Operators

We can generalise the operations for qubits ($d = 2$) to $\forall n \in \mathbb{N}, d = n$. For $d = 2$, we know that:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{61}$$

Generalising the Pauli Z, X to arbitrary d, the Z is called the clock operator, while X is called the shift operator. The clock operator has the form:

$$Z = \text{diag}\left(1, e^{i\frac{2\pi}{d}}, \cdots e^{i\frac{(d-1)\cdot\pi}{d}}\right) = \sum_{j=0}^{d-1} \omega_j |j\rangle\langle j|, \tag{62}$$

where $\omega_j = e^{i\frac{2\pi j}{d}}$. By convention, the shift operator has the form:

$$X = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} = \sum_{i=0}^{n-1} |i+1\rangle\langle i|, \tag{63}$$

where $|d\rangle = |0\rangle$. In both cases, we have the restriction that $Z^d = X^d = \mathbb{I}$. Comparing to $d = 2$, hermicity is not preserved for higher dimensions $d \geq 3$. However, these operators are always unitary. Consequently, in contrast to the case where $d = 2$, the generalised Pauli matrices are not the elements of the real Lie algebra $\mathfrak{su}(3^k)$, whose elements are traceless Hermitian matrices of size $3^k \times 3^k$.

5.1.1 Commutation of X, Z

We know generally the commutation relation for the d-dimensional Z, X operators:

$$ZX = \omega XZ, \tag{64}$$

where $\omega = e^{i\frac{2\pi}{d}}$. Since we can write any of the Pauli matrices (on one qudit) in the form $Z^a X^b$ up to some phase. Thus, we can show that arbitrary Pauli matrices commute. For some $a, b, c, d \bmod(n)$, we have:

$$\begin{aligned} \left[Z^a X^b, Z^c X^d\right] &= Z^a X^b Z^c X^d - Z^c X^d Z^a X^b \\ &= (\omega^{-b})^c Z^a Z^c X^b X^d - (\omega^{-d})^a Z^c Z^a X^d X^b \\ &= [(\omega^{-b})^c - (\omega^{-d})^a] Z^{a+c} X^{b+d}, \end{aligned} \tag{65}$$

where we discard the coefficient if it's non-zero. Thus, the commutator is either $\mathbb{0}$ or an orthogonal Pauli matrix. This informs our algorithm for computing $\dim(A_k)$. Additionally, this concept is easily extendable to multiple qudits, as is accounted for in the algorithm.

5.2 Generalisation of T_k

Generalisation of the Hadamard H to the d-dimensional qudit is:

$$H = \frac{1}{\sqrt{d}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^{d-1} & \omega^{2(d-1)} & \cdots & \omega^{(d-1)^2} \\ 1 & \omega^{d-2} & \omega^{2(d-2)} & \cdots & \omega^{(d-1)(d-2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega & \omega^2 & \cdots & \omega^{d-1} \end{bmatrix} = \sum_{j,k=0}^{d-1} (\omega_{d-j})^k \, |j\rangle\langle k|, \tag{66}$$

where $\omega_{d-j} = e^{i\frac{2\pi(d-j)}{d}}$. The generalisation of CZ is:

$$CZ = \sum_i |i\rangle\langle i| \otimes Z^i. \tag{67}$$

The tensor network representation of these operators remains the same, and the generalised CZ can be represented as the generalised H. However, for the generalisation of the S matrix, we need to ensure the dual-unitarity of the operator T_k. To achieve this, we choose any diagonal matrix within the qudit Clifford group as our S matrix.

For instance, when we consider the case of qutrits where $d = 3$, our choice of S is given by I, Z, Z^2, $\mathrm{diag}(1, 1, w_1)$ and all permutations, and $\mathrm{diag}(1, 1, w_2)$ and all permutations. Using the above generalisations, we can test out different choices of S. The local rules of conjugation by T_k for each choice of S is listed in Appendix 9.6.

Finally, we consider the generalised basis of measurement. Note that the Lie algebra for $\mathfrak{su}(d^k)$, with $d > 2$, is generated by the Gell-Mann matrices rather than the Pauli matrices (see Appendix 9.5). Thus, instead of measuring rotations about X, we measure the exponentiation of linear combinations of Gell-Mann matrices. For such choices of d, we cannot easily move over the H operator through the measurement (as done in Equation 47). Thus, we must account for this in our measurement by measuring in the basis:

$$\{He^{iA}H|0\rangle, \ldots, He^{iA}H|d-1\rangle\}, \tag{68}$$

where:

$$A = \theta_1\lambda_{\sigma(1)} + \cdots + \theta_{d-1}\lambda_{\sigma(d-1)}, \tag{69}$$

for angles θ_j corresponding to unique diagonal Gell-Mann matrices $\lambda_{\sigma(j)}$ (from Equation 98).

In the qutrit case where $d = 3$, we measure in the basis:

$$\{He^{i(\theta_1\lambda_3+\theta_2\lambda_8)}H|0\rangle, He^{i(\theta_1\lambda_3+\theta_2\lambda_8)}H|1\rangle, He^{i(\theta_1\lambda_3+\theta_2\lambda_8)}H|2\rangle\}, \tag{70}$$

given diagonal Gell-Mann matrices λ_3, λ_8 for $d = 3$ from Equation 95.

5.2.1 Correspondence Between the Minimal Lie Algebra and Pauli Matrices

The Gell-Mann matrices form a basis over the vector space of traceless Hermitian matrices, which also spans the Lie algebra of $\mathfrak{su}(d^k)$ (see Appendix 9.5). More precisely, these vectors are real, in the sense that the coefficients allowed in a linear superposition are real numbers. To find the minimal Lie subalgebra $\dim(\mathcal{A}_k)$ for qudits obtainable from our MBQC scheme, we repeatedly apply the Lie bracket on $O_k(j) \in \mathcal{O}_k$ and find the set of linearly independent elements that are eventually closed under the lie bracket; $\mathcal{A}_k$ is then the (real) linear span of this set. As mentioned before, if $\mathcal{A}_k$ is found to be equal to $\dim \mathfrak{su}(d^k)$ for a d-dimensional qudit, then we will be able to generate any unitary operator in $\mathrm{SU}(d^k)$ using the MBQC protocol.

Although the aforementioned method for determining $\mathcal{A}_k$ is conceptually straightforward, it suffers from computational inefficiency. The primary reason for this issue is that, after commutating two elements $O_k(j), O_k(l)$, we must implement the Gram-Schmidt process to determine

the linearly-independent matrix relative to all previously discovered generators. This procedure rapidly grows in complexity with increasing values of d and k.

To overcome this challenge, we can employ the use of generalised Pauli matrices, which although not Hermitian in general, offer a straightforward commutation relation. By complexifying the real Lie algebra of interest, we can transform it into a complex Lie algebra with complex coefficients. In this complex Lie algebra, the generalized Pauli matrices form a basis, enabling us to obtain the minimal complex Lie algebra $\mathcal{B}_k$ by closing any complex linear combination of $O_k(j)$ and $O_k(l)$ under commutation.

Interestingly, the dimension of the minimal real Lie subalgebra $\mathcal{A}_k$, obtained by closing $O_k(j)$ and $O_k(l)$ under commutation, is the same as the dimension of $\mathcal{B}_k$ considered over the complex field.[6] In other words, we can express this as:

$$\dim_{\mathbb{R}}(\mathcal{A}_k) = \dim_{\mathbb{C}}(\mathcal{B}_k). \tag{71}$$

Moreover, we can generate other complex linear combinations of $O_k(j)$ and $O_k(l)$, such as $O_k(j) + O_k(l)$ and $O_k(j) - O_k(l)$, to obtain $\mathcal{B}_k$.

In our case, we know that the clock operator Z is a complex linear combination of the Gell-Mann matrices (for qutrits):

$$Z = \frac{1}{4}(\sqrt{3}\lambda_8 - 3\lambda_3) + \frac{1}{2}i \sin\frac{2\pi}{3}(\sqrt{3}\lambda_8 - \lambda_3) \tag{72}$$

and similarly for Z^2, implying that λ_3, λ_8 are themselves complex linear combinations of Z, Z^2. We then notice that $O_k(j)$'s are formed by conjugating λ_3, λ_8 with multiple applications of T_ks; this means in the complexified version we can consider Z, Z^2 with T_k. The key insight to be made in the latter is that T_k is a Clifford operation, such that $T^i Z T^{i\dagger}$, $T^i Z^2 T^{i\dagger}$ is again a generalised Pauli string, whose commutation relations with other Pauli strings are simple to compute.

Thus, to find $\mathcal{B}_k$, we have a computationally efficient algorithm utilizing the Pauli group structure. We employ this algorithm in the next section, determining $\mathcal{B}_k$, which in turn allows us to determine $\dim(\mathcal{A}_k)$.

[6] See https://math.stackexchange.com/questions/4669307/minimal-lie-subalgebra-generated

6 Algorithm for Finding dim($\mathcal{A}_k$)

We aim to find the size of the set $\mathcal{B}_k$, which is the set of matrices determined by closing the set $\mathcal{O}_k$ under commutation.[7] To achieve this, we first input a set of orthonormal Pauli matrices O_k belonging to the generalised Pauli group.

For our algorithm, we do not consider the explicit matrix representation of the Pauli matrices. Since Pauli matrices can be represented in terms of:

$$Z_1^{i_1} \cdots Z_n^{i_n} X_1^{j_1} \cdots X_n^{j_n}, \tag{73}$$

up to some global phase for some $i_1, \ldots i_n, j_i, \ldots, j_n \in \{0, \ldots, n-1\}$. Thus, a list representation of the i_k and j_k indices is isomorphic to the matrix representation of the Pauli matrices (up to some global phase) (see Appendix 9.3.2). For example, $Z_1 X_2$ for two qubits can be represented as $((1,0),(0,1))$, where the left array corresponds to the upper indices for the Z matrix at each qudit, and the right array corresponds to the upper indices for the X matrix. This form (compared to the matrix representation) will greatly decrease the computation time, and is easily generalisable.

We then use the commutation laws described in Subsection 5.1.1 to compute whether commutation will create a new Pauli matrix. Commutation will create a non-zero matrix if the sum of $-bc + da$ (notation from Subsection 5.1.1) for all qudits, after then taking $(\mathrm{mod}\, n)$, is non-zero. We check if the newly created Pauli matrix is already contained in the set of matrices.

The below pseudocode is an implementation of an algorithm which finds dim($\mathcal{B}_k$) (equivalently, dim($\mathcal{A}_k$)) given $\mathcal{O}_k$:

```
ortho = O_k
new_ortho = empty list
size = number of qudits
dim = d
loop = True
While (loop == True)
   for M_1 in ortho
      for M_2 in ortho
         M_c = null list containing two lists of length size
         coefficient = 0
         for i from 0 to dim-1
            a = upper index of Z for qudit i in M_1
            b = upper index of X for qudit i in M_1
            c = upper index of Z for qudit i in M_2
            d = upper index of X for qudit i in M_2
            upper index of Z for qudit i in M_c = a + c (mod dim)
            upper index of X for qudit i in M_c = b + d (mod dim)
            Add -bc + ad to coefficient
         If (coefficient (mod dim) = 0) is not true
            if (M_c in ortho or new_ortho) is not true
               Add M_c to new_ortho
   if new_ortho is empty
      loop = False
   ortho = ortho + new_ortho
   new_ortho = empty list
dim(A_k) = size of ortho
```

[7] We have chosen the diagonal Pauli matrices rather than the Gell-Mann matrices based on the idea of complexification

6.1 Algorithm Efficiency

In order to improve the efficiency of our algorithm, we can reduce the size of the commutator set at each iteration. After adding a new orthonormal matrix to the set, we only need to iterate through the commutators of this new matrix with the whole set of orthonormal matrices, since we have already iterated through the commutators of the previous matrices in the set.

Moreover, to further optimise the algorithm, we can choose to commute only a subset of the newly obtained matrices from the commutator set `new_ortho`. Instead of iterating through all the newly obtained matrices, which could be hundreds or even thousands, we can select a smaller subset, such as the first 10 matrices. This significantly reduces computation time while still giving us a lower bound for the dimension of $\mathcal{A}_k$.

6.2 Implementation for $d = 2$

We extend the results from [6] using our algorithm, from $k = 7$ to $k = 8, 9$: We also record the time taken to attain the $\dim(\mathcal{A}_k)$ results for different k values. We choose only 10 new Pauli matrices from `new_ortho` to commute with the elements in `ortho`.

k	7	8	9
$\dim(\mathcal{O}_k)$	24	28	60
$\dim(\mathcal{A}_k)$	8256	32896	262143
time(s)	4	15	146

For the choice $k = 9$, we know that $\dim(\mathcal{A}_k) = \mathfrak{su}(2^k)$, so we can implement any quantum algorithm for 9 logical qubits.

6.3 Implementation for $d = 3$

Based on Section 5.2, we can determine $\mathcal{O}_k$ for choices of k. Measurement of physical qubits is in the basis $\{e^{-i\theta X_1}|s\rangle\}$. First, we take $S = I, Z, Z^2$ and show the values from $k = 3$ to $k = 7$. Note that the times can be improved with different choices of the number of new matrices to commute.

k	3	4	5	6	7
$\dim(\mathcal{O}_k)$	16	20	24	28	32
$\dim(\mathcal{A}_k)$	728	6560	59048	531440	4782968
time(s)	< 1	1	19	180	2740

where for all given k, $\dim(\mathcal{A}_k) = \dim \mathfrak{su}(3^k) = 9^k - 1$. Note that the computation time scales exponentially with k as $\mathfrak{su}(3^k) = 9^k - 1$ scales exponentially with k.

For $S = \mathrm{diag}(1, 1, w_1)$ and $\mathrm{diag}(1, 1, w_2)$, we have (without the time to compute):

k	3	4	5	6	7
$\dim(\mathcal{O}_k)$	60	80	72	728	480
$\dim(\mathcal{A}_k)$	728	6560	59048	531440	4782968

Under preliminary testing, it seems that choice of S does not affect $\dim(\mathcal{A}_k)$. However, the choice of S affects p_k, which in turn may change how we decompose our desired logical unitary. Depending on the unitary we want to generate, different choices of S may affect the number of qudits N required.

In addition, given period p_k for the generalised T_k, our values of $\dim(\mathcal{O}_k)$ are actually:

$$\dim(\mathcal{O}_k) = 2p_k. \tag{74}$$

This is because we have access to the following exponents of Gell-Mann matrices:

$$\mathcal{O}_k = \{e^{i\theta\lambda_3}, T_k^\dagger e^{i\theta\lambda_3} T_k, \ldots, \left(T_k^\dagger\right)^{p_k-1} e^{i\theta\lambda_3} T_k^{p_k-1}, e^{i\theta\lambda_8}, T_k^\dagger e^{i\theta\lambda_8} T_k, \ldots, \left(T_k^\dagger\right)^{p_k-1} e^{i\theta\lambda_8} T_k^{p_k-1}\}. \tag{75}$$

For our algorithm, we consider the equivalent complexified Gell-Mann matrices, which gives us the following Pauli matrices:

$$\mathcal{O}_k = \{Z_1, T_k^\dagger Z_1 T_k, \ldots, \left(T_k^\dagger\right)^{p_k-1} Z_1 T_k^{p_k-1}, Z_1^2, T_k^\dagger Z_1^2 T_k, \ldots, \left(T_k^\dagger\right)^{p_k-1} Z_1^2 T_k^{p_k-1}\}, \quad (76)$$

where it is easily shown that $\dim(\mathcal{O}_k) = 2p_k$.

7 Conclusion

We have discovered a new family of universal one-dimensional resource states in systems with higher local dimensions (qudits). These resource states possess two crucial properties. Firstly, the entangling operators used to generate the state are dual-unitary, establishing a correspondence between the physical and logical operators. Secondly, the operator is Clifford, which means that measurement errors propagate controllably, and can be corrected in an efficient manner classically.

Interestingly, there are many equivalent resource states for systems beyond qubits (choices of S), and each state has a different period (or clock cycle length) p_k. This period sets the clock cycle of the evolution and determines the number of physical resources required to perform one basic unitary $e^{iO_k(j)}$ where $O_k(j) \in \mathcal{O}_k$. A smaller clock cycle is advantageous, as it reduces the number of physical resources required. However, the clock cycle limits the cardinality of $\mathcal{O}_k$, the number of such $O_k(j)$ accessible under evolution. It may be necessary to perform many cycles to accurately reach an element $A \in \mathcal{A}_k$, the Lie algebra formed by closing $\mathcal{O}_k$ under the Lie braket. Therefore, there is a trade-off between the resources required for implementing basic unitaries and those required for implementing the target unitarity from these building blocks.

Additionally, we utilised the complexification of the Lie algebra to establish a relationship between the Lie algebra and Gell-Mann matrices. This helps us reduce the computational overhead required to calculate $\mathcal{A}_k$ for higher dimensions.

In subsequent research, identifying algorithms capable of decomposing desired unitaries to choices of measurement within the MBQC scheme described would be an intriguing avenue for investigation. Furthermore, exploring the efficiency of various resource states concerning the number of necessary physical qudits could prove valuable. Such an investigation may illuminate the practicality of implementing these resource states in experimental quantum systems.

References

[1] Preskill, J. (2023). Quantum computing 40 years later (arXiv:2106.10522; Version 3). arXiv. https://doi.org/10.48550/arXiv.2106.10522

[2] Miller, J. E. (2017). Measurement-Based Quantum Computation and Symmetry-Protected Topological Order. https://digitalrepository.unm.edu/phyc_etds/149

[3] Browne, D. E., & Briegel, H. J. (2006). One-way Quantum Computation—A tutorial introduction (arXiv:quant-ph/0603226). arXiv. https://doi.org/10.48550/arXiv.quant-ph/0603226

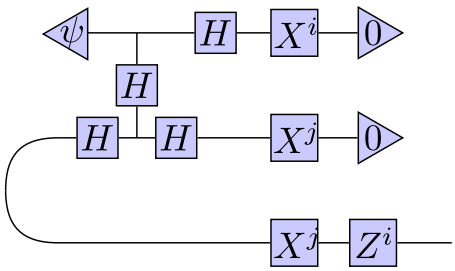
[4] Bridgeman, J. C., & Chubb, C. T. (2017). Hand-waving and Interpretive Dance: An Introductory Course on Tensor Networks. Journal of Physics A: Mathematical and Theoretical, 50(22), 223001. https://doi.org/10.1088/1751-8121/aa6dc3

[5] Raussendorf, R., Browne, D. E., & Briegel, H. J. (2002). The one-way quantum computer—A non-network model of quantum computation. Journal of Modern Optics, 49(8), 1299–1306. https://doi.org/10.1080/09500340110107487

[6] Stephen, D. T., Ho, W. W., Wei, T.-C., Raussendorf, R., & Verresen, R. (2022). Universal measurement-based quantum computation in a one-dimensional architecture enabled by dual-unitary circuits (arXiv:2209.06191). arXiv. http://arxiv.org/abs/2209.06191

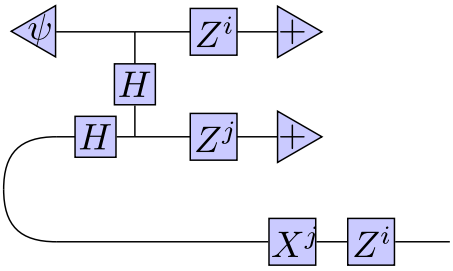
9 Appendix

9.1 Tensor Network Manipulations for Teleportation



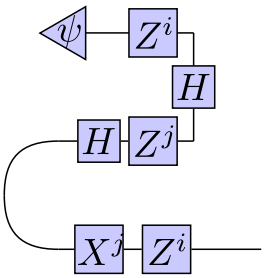
(77)

becomes:



(78)

Using Equation 13, the above diagram becomes:



(79)

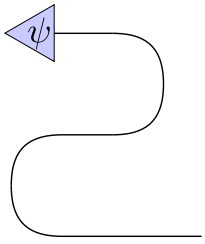
and since:

$$Z^i X^j H Z^j H Z^i = Z^i X^j X^j H H Z^i = Z^i X^j X^j Z^i = Z^i Z^i = \mathbb{I}, \tag{80}$$

where:

$$i, j \in \{0, 1\} \implies Z^{2i}, X^{2j} = \mathbb{I}, \tag{81}$$

such that the operations will simplify to $\mathbb{I}$. Thus, the final form of the tensor network looks like:

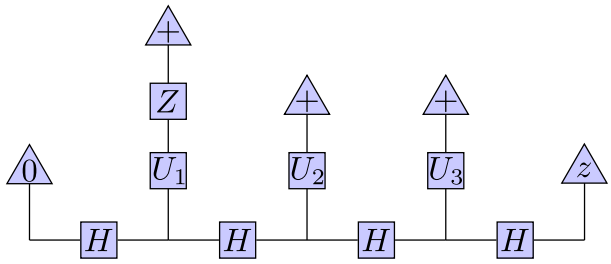


(82)

as desired.

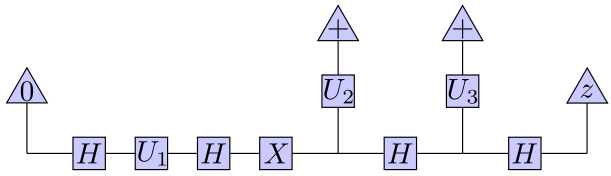
9.2 Tensor Network Manipulations for 1D Cluster State

We have:



(83)

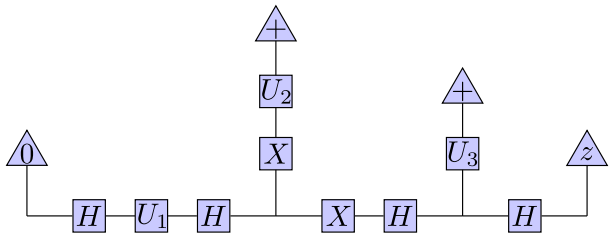
Since Z commutes with U_1, and both Z, U_1 are diagonal matrices, the above diagram is equivalent to:



(84)

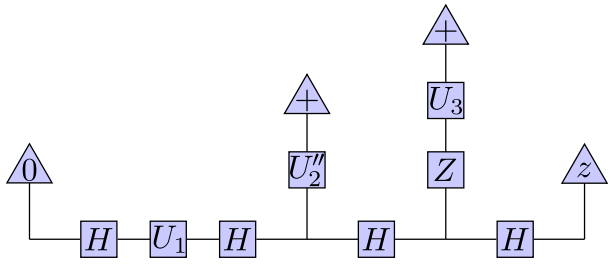
where we used Equation 13 to remove the $|+\rangle$ state. Additionally, we move the Z operator through the H operator to get X.

To move the X operator to the third and fourth qubit, we use the identity from Equation 15. This gives us the diagram:



(85)

which simplifies to:



(86)

where $U_2'' = e^{-i(\theta_2/2)Z}$ and we move the X operator over the H. Finally, we can absorb the Z into U_3 to get $U_3' = e^{i(\theta_3/2+\pi/2)Z}$, which gives us the desired diagram.

9.3 Pauli Group and Clifford Group for Qudits

9.3.1 Pauli Group and Clifford Group for Qubits

For n qubits, the Pauli group is given by:

$$\mathcal{P}_n = \left\{ e^{i\theta\pi/2} \sigma_{j_1} \otimes \cdots \otimes \sigma_{j_n} \right\}, \tag{87}$$

where $\theta \in \{0, 1, 2, 3\}$ and $j_1, \ldots, j_n \in \{0, 1, 2, 3\}$. Additionally, we know that:

$$\sigma_0, \sigma_1, \sigma_2, \sigma_3 = \mathbb{I}, \sigma^x, \sigma^y, \sigma^z. \tag{88}$$

Informally, the Pauli group is the set of tensor products of n operators, which are either the single-qubit Pauli operators (X, Y, Z) or single-qubit identity. The whole operator is multiplied by the fourth roots of unitary.

For n qubits, the Clifford group is:

$$\mathcal{C}_n = \{U \in \mathrm{SU}(2^n) : UP_nU^\dagger \in P_n\}, \tag{89}$$

where $\mathrm{SU}(2^n)$ is the special unitary group of $2^n \times 2^n$ unitary matrices. Thus, elements of the Pauli group under conjugation of elements of the Clifford group will still belong to the Pauli group. We know that the Hadamard operator H, phase operator $S = \sqrt{Z}$, and the Controlled-Z gate CZ together are the generators of the Clifford group.

See https://en.wikipedia.org/wiki/Clifford_gates

9.3.2 Generalisation to Qudits

The generalisations of the Pauli operators and Clifford operators are given in Section 5.1 and Section 5.2.

Given $d > 2$, for n qudits, we can choose the d-dimensional Pauli group as:

$$\mathcal{P}_n = \left\{ e^{i2\pi n/d} Z_1^{j_1} X_1^{k_1} \cdots Z_n^{j_n} X_n^{k_n} \right\}, \tag{90}$$

where $\theta \in \{0, 1, \ldots, n-1\}$ and $j_1, \ldots, j_n, k_1, \ldots, k_n \in \{0, \ldots, n-1\}$. Note that the global phase are just the powers of ω. Though there are multiple ways to generalise the Pauli operators, our choice of the Pauli group comes from generalising the Z, X operators (as the other single qudit Pauli operators can be expressed as multiples of Z, X up to some phase).

For n qudits, the Clifford group becomes:

$$\mathcal{C}_n = \{U \in \mathrm{SU}(d^n) : UP_nU^\dagger \in P_n\}. \tag{91}$$

9.4 Lie Algebra for Qubits

In quantum mechanics, Lie algebras and Lie groups are important mathematical structures which characterise the time evolution of a quantum state. The set of unitary operators that act on a quantum system constitutes a Lie group (special unitary group SU(2) for the single qubit case), which are generated by a set of operators in a Lie algebra ($\mathfrak{su}(2)$, the Pauli matrices). Explicitly, $U = e^{-iHt}$ is an element of a Lie group, where H is an element of a Lie algebra. Lie groups exhibit closure under multiplication, while Lie algebras exhibit closure under commutation. Using the Baker-Campbell-Hausdorff formula, we know that:

$$\forall A, B \in \mathfrak{su}(2),\ e^A e^B = e^{A+B+\frac{1}{2}[A,B]+\cdots}. \tag{92}$$

Using this formula, we can relate multiplications of the Lie group with commutations in the Lie algebras.

Importantly, we know that:

$$e^{i\theta A}e^{i\phi B}e^{-i\theta A}e^{-i\phi B} = e^{i\theta A+i\phi B-\frac{1}{2}\theta\phi[A,B]+\cdots}e^{-i\theta A-i\phi B-\frac{1}{2}\theta\phi[A,B]+\cdots} = e^{\theta\phi[A,B]+\cdots}, \tag{93}$$

where A, B commute such that $[A, B] = \mathbb{0}$ or $[A, B]$ forms an orthogonal element in the Lie algebra. Thus, if we have full control in measuring angles θ, ϕ about the Lie algebras A, B, we also have control over the angle $\xi = \theta\phi$ about the Lie algebra $[A, B]$.

As we briefly mentioned in Section 3.1.1, for one qubit, the Euler decomposition can generate arbitrary elements of the Lie group. Here, the Pauli Z, X are the generators of the Lie algebra.

To generate the Lie algebras for the for k qubits, we consider the Tensor product of single qubit Paulis and identities (minus the k-qubit identity). Thus, the number of generators of the Lie algebra becomes:

$$\dim \mathfrak{su}(2^k) = 4^k - 1. \tag{94}$$

See https://quantumcomputing.stackexchange.com/questions/31791/link-between-quantum-computing-and-lie-theory

9.5 Lie Algebra for Qutrits and Qudits

For $\mathfrak{su}(3)$, the generators of the Lie algebra are the Gell-Mann matrices:

$$\lambda_1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \lambda_2 = \begin{pmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \lambda_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \lambda_4 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$
$$\lambda_5 = \begin{pmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{pmatrix}, \lambda_6 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \lambda_7 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{pmatrix}, \lambda_8 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{pmatrix}, \tag{95}$$

and since the Gell-Mann matrices are not Pauli, the Gell-Mann matrices are not normalised by T_k.

According to https://mathworld.wolfram.com/GeneralizedGell-MannMatrix.html, for the generalisations of the Gell-Mann matrices $\mathfrak{su}(d)$, denote $E_{j,k}$ as the $d \times d$ matrix with 1 at the j,k-th entry, and 0 elsewhere. The Gell-Mann matrices can a collection of three types of matrices.

The first are the real symmetric matrices:

$$E_{j,k} + E_{k,j}, \tag{96}$$

ranging over the index j,k where $1 \leq j < k \leq d$. .

The second are the imaginary antisymmetric matrices:

$$-i(E_{j,k} + E_{k,j}), \tag{97}$$

ranging over the index j,k where $1 \leq j < k \leq d$.

The third are the diagonal real matrices:

$$\sqrt{\frac{2}{k(k+1)}} \left(\sum_{j=1}^{k} E_{j,j} - kE_{k+1,k+1} \right), \tag{98}$$

ranging over k where $1 \leq k \leq d-1$

In general, the cardinality of the Lie algebra corresponding to k d-dimensional qudits is:

$$\mathfrak{su}(d^k) = d^{2k} - 1. \tag{99}$$

9.6 Choice of Qutrit S and Evolution

We measure the physical qubits in the basis $\{e^{-i\theta X_1}|s\rangle\}$. Choice of $S = \mathrm{diag}(1, 1, \omega_1)$ and permutations will generate:

$$\begin{aligned} T_k X_i T_k^\dagger &= Z_i^2, \\ T_k Z_i T_k^\dagger &= \begin{cases} Z_1^2 X_1 Z_2^2 & i = 1, \\ Z_{i-1}^2 Z_i^2 X_i Z_{i+1}^2 & 1 < i < k, \\ Z_{k-1}^2 Z_k^2 X_k & i = k. \end{cases} \end{aligned} \tag{100}$$

Choice of $S = \mathrm{diag}(1, 1, \omega_2)$ and permutations will generate:

$$\begin{aligned} T_k X_i T_k^\dagger &= Z_i^2, \\ T_k Z_i T_k^\dagger &= \begin{cases} Z_1 X_1 Z_2^2 & i = 1, \\ Z_{i-1}^2 Z_i X_i Z_{i+1}^2 & 1 < i < k, \\ Z_{k-1}^2 Z_k X_k & i = k. \end{cases} \end{aligned} \tag{101}$$

Choice of $S = I, Z, Z^2$ will generate:

$$\begin{aligned} T_k X_i T_k^\dagger &= Z_i^2, \\ T_k Z_i T_k^\dagger &= \begin{cases} X_1 Z_2^2 & i = 1, \\ Z_{i-1}^2 X_i Z_{i+1}^2 & 1 < i < k, \\ Z_{k-1}^2 X_k & i = k. \end{cases} \end{aligned} \tag{102}$$