A Tensor Network Implementation of the Holographic Pentagon Code

Esteban V. Prado

University of California Berkeley

(Dated: December 2023)

Abstract

This work was done in the context of the course Quantum Information Science and Technology (CS191) at UC Berkeley. In this work, I used Python, Numpy and the TensorNetwork Library from Google, to code a Tensor Network of the holographic pentagon code (HPC) proposed in [1]. The main results are 1. a one-layer HPC 2. with the capability of pushing a bulk operator in the central pentagon to the boundary, and 3. the *Holographic Code Library*, a collection of functions and objects that abstract away many of the tensor network details. Here, you can find the Holographic Code Library and a well documented Jupyter Notebook with a demonstration.

CONTENTS

Contribution Statement: I confirm that this project is my own work only and I have documented all sources and materials used.

1. MOTIVATION

It is well known that General Relativity (GR) and Quantum Mechanics (QM) are inconsistent. The field which tries to reconcile this two theories is Quantum Gravity (QG), and one way to study quantum gravity is through the AdS/CFT Correspondence. In [1] a toy model of an of an AdS/CFT Correspondence is presented. The toy model has many limitations, but the hope is that through this toy model, we can still gain valuable intuition about QG and inform our decisions of in what direction to keep searching.

2. BACKGROUND THEORY

This section briefly explains the necessary background knowledge to understand this work. To learn more about the theory of the AdS/CFT Correspondence and the relation with Quantum Error Correcting Codes, I can recommend reading [2].

2.1. Ads/CFT Correspondence

The AdS/CFT Correspondence is a conjectured relationship between two kinds of physical theories: on the one hand a Quantum Gravity (QG) on a (d+1)-dimensional asymptotically-Anti de Sitter space, and on the other hand, a d-dimensional Conformal Field Theory (CFT). The Quantum Gravity is formulated in terms of a String Theory or M-Theory, and the CFT can be for example Quantum Field Theory. The AdS is also called the bulk and the CFT is called boundary. And the reason why the world "hologram" is used, is that the boundary is a d-dimensional space which stores information about a (d+1)-dimensional space. For example a 2D picture of a 3D room can be considered a hologram of the room. Figure 1 summarizes this section.

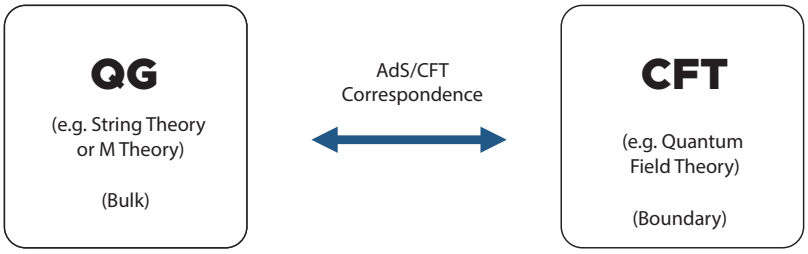

FIG. 1: Summary of the AdS/CFT Correspondence

2.2. Holographic Pentagon Code

The Holographic Pentagon Code is shown in Figure 2. The red circles represent the states of the Quantum Gravity (AdS). Since the red circles are inside the geometry, we refer to them collectively as the bulk-state. Similarly, the white circles represent the states of the Quantum Field Theory (CFT), since the white circles are on the border of the geometry, we refer to them collectively as the boundary state. Each circle is a two level system, so we refer to them as qubits. The pentagons can be understood as a 3 qubit gate, which is derived from the 5-qubit quantum error correcting code. Figure 3 explains how to understand this equivalence. The way in which the gate is derived is explained in section 4.2.

2.3. What is Operator Pushing

The idea behind operator pushing is that we map an operator from the bulk to the boundary. This would be equivalent to doing a perturbation on the Quantum Gravity and finding an equivalent perturbation in the Quantum Field Theory. A more appropriate name might be "operator mapping".

2.4. Tensor Networks

Tensors are a generalization of vectors and matrices. A scalar is a rank 0 tensor, a vector is a rank 1 tensor and a matrix is a rank 2 tensor. A rank n tensor has n indices and is denoted like so:

$$T_{i_1,i_2,\cdots,i_n}$$

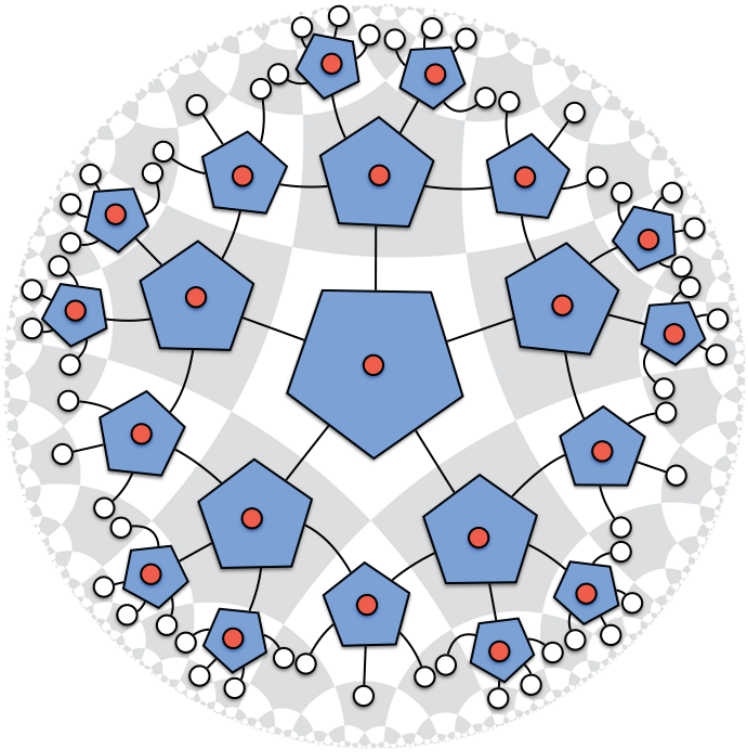

FIG. 2: Holographic Pentagon Code with 3 layers. Adapted from [1].

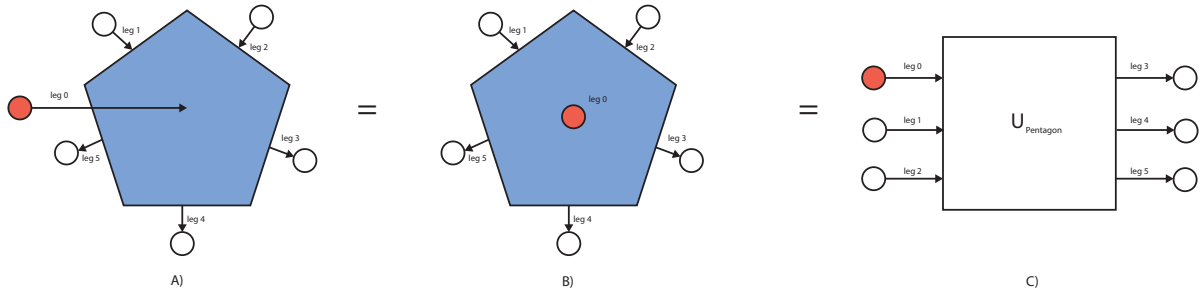

FIG. 3: Summary of the AdS/CFT Correspondence: A) Shows the pentagon tensor, which has six legs contracted with qubits. B) Shows the usual representation, where the leg 0 contracted with the red qubit is above the pentagon plane. C) Shows the equivalent quantum circuit representation.

In the graphical representation, a tensor is described by a node and n legs. The number of legs is equal to the number of indices. The shape of the node is often a circle, a square or a triangle. But there is no convention for the meaning of the shape. It can be

used to differentiate the interpretation of different tensors. In our case, circles represent qubit states, squares represent operators and the pentagons represent 3-qubit gates.

A rank n tensor has n legs, which are graphically represented by a line, coming out of the node. When the leg of two tensors are connected, we say that they are *contracted*. Contraction is defined by the following operation:

$$C_{\alpha\gamma} = \sum_{\beta=1}^{D} A_{\alpha\beta} B_{\beta\gamma}$$

For example, to perform vector-matrix multiplication we connect/contract the only leg of a rank 1 tensor with a leg of a rank 2 tensor as shown in Figure 4. The resulting tensor has only one leg, indicating that it is a vector as expected.

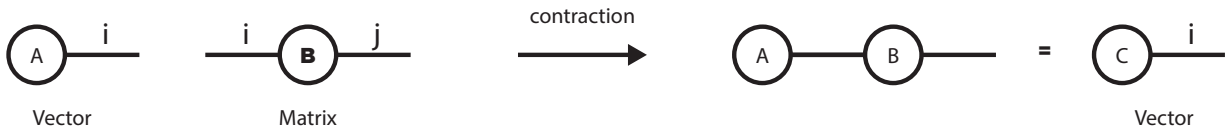

FIG. 4: Vector-Matrix multiplication in the tensor network notation.

But why use tensor networks? As motivated in 3.3 in [3]: Consider a quantum many-body wave-function. It is intuitive that the entanglement structure varies depending on the dimensions of the system or the position of the bodies. It is also influenced by subtler factors like the state's criticality and correlation length. This information would not be described in coefficients of a wave function. In contrast, a Tensor Network stores information in terms of a network of quantum correlations.

In the specific case of the HPC, because the pentagons are perfect tensors, there exists an equivalent quantum circuit representation of a pentagon code, but this representation lacks any information of where the particles are located in space.

If you are interested in learning more about tensor networks, Appendix A includes a list of resources that I compiled.

2.5. The 5-Qubit Quantum Error Correcting Code

The 5-Qubit Quantum Error Correcting code maps the states $|0\rangle$ and $|1\rangle$ into a code-word of 5 qubits. For example, $|0\rangle$ is mapped to:

$$|0\rangle \rightarrow |00000\rangle + |10010\rangle + |01001\rangle + |10100\rangle + |01010\rangle - |11011\rangle - |001100\rangle - |11000\rangle \\ - |11101\rangle - |00011\rangle - |11110\rangle - |01111\rangle - |10001\rangle - |01100\rangle - |10111\rangle + |00101\rangle$$

We call the original qubit $|0\rangle$ the logical qubit. Similarly, we call the 5 mapped qubits, the physical qubits.

3. MAIN RESULTS

The main results are:

1. A one-layer Holographic Pentagon Code,
2. with the capability of pushing an operator in the central bulk qubit to the boundary qubits,
3. and the *Holographic Code Library*, a collection of functions and objects that abstract away many of the tensor network details.

Here, you can find the Holographic Code Library and a well documented demonstration in a Jupyter Notebook. I would highly encourage you to check out this notebook if you are interested in concrete examples of the results.

4. METHODS

4.1. Implementation of Tensor Networks

To implement the tensor network I used Python, Numpy and the TensorNetwork Library from Google.

4.2. Obtaining the Pentagon Gate

To derive the matrix which represents the pentagon gate, we label the the logical qubit as qubit zero: $|q_0\rangle$ and we label the physical qubits as qubits one to five: $|q_1 \cdots q_5\rangle$. The state

is then $|\psi\rangle = |q_0\rangle \otimes |q_1 \cdots q_5\rangle = |q_0 \cdots q_5\rangle$. We now interpret qubits $|q_0 q_2 q_2\rangle$ as the input and qubits $|q_3 q_4 q_5\rangle$ as the output. This interpretation is consistent with the arrow directions in Figure 3.

To get the output for a given input, we project the input vector on the state. For example to get the output $|q_3 q_4 q_5\rangle$ for the input $|000\rangle_{123}$, we compute:

$$|q_3 q_4 q_5\rangle = \langle 000|_{123} |\psi\rangle$$

.

Column j of the matrix which represents the pentagon gate is the output vector $|q_3 q_4 q_5\rangle$ for the input vector $|binary(j)\rangle$. For a 3-qubit gate, $j = \{0, ..., 7\}$. One can confirm this matrix is unitary, by verifying in python that $U^\dagger U = U U^\dagger = I$.

4.3. Operator Pushing

Operator pushing consists of 4 steps:

1. **Promote a bulk operator to a 3-qubit gate:** $O \to O \otimes I \otimes I$.
2. **Get pushed operator:** We use the property that $UO = UOI = UOU^\dagger U = (UOU^\dagger)U = O_{pushed}U$.
3. **Factorize Pushed Operators:** An algorithm (explained in section 4.4) finds 3 operators A, B, and C, such that:$A \otimes B \otimes C = O_{pushed}$.
4. **Keep pushing operators until we reach the boundary:** If A, B and C are connected to another pentagon, we successively push them through individual pentagons, until we reach the boundary. We know how many times we need to push until we reach the boundary, because the number of layers is defined in the code.

The first three steps of this process are shown in Figure 5.

4.4. Factoring Operators

Factoring an operator means to find a disjoint set of one-qubit operators such that the tensor product of such operators is equal to the original operator. For example, a 3-qubit

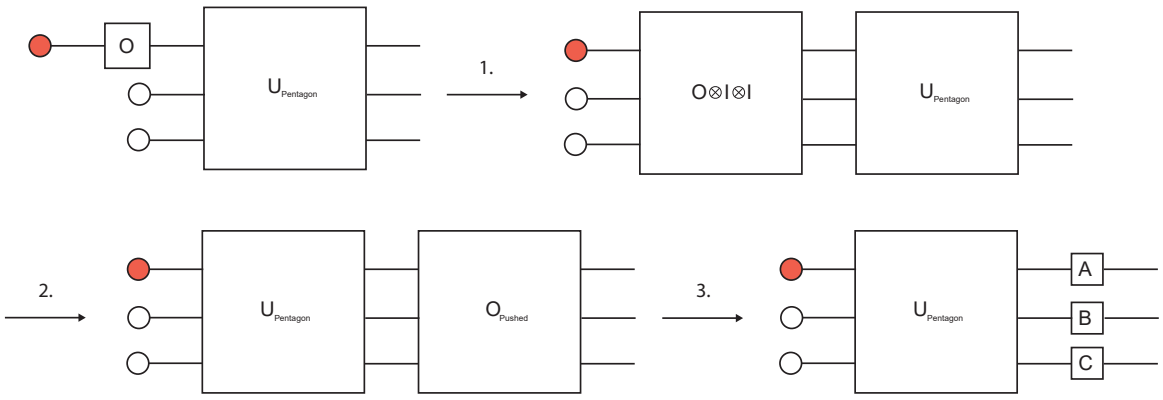

FIG. 5: The process of operator pushing: 1. promote bulk operator to a 3-qubit gate, 2. get pushed operator, and 3. factorize pushed operator.

gate O represented by an 8x8 matrix, can be factorized into 3 1-qubit gates A, B and C, each represented by 2x2 matrix: $O = A \otimes B \otimes C$. Generally speaking, it is not guaranteed that an operator is factorizable. In our case, it is possible because the pentagon gate is a perfect tensor [1].

The *Holographic Code Library* implements an algorithm which factorizes an operator O into two operators A and B, where A is a 2x2 matrix. To find the three operators, the algorithm is applied again on B. The algorithm divides the matrix O into 4 blocks and then finds the greatest common denominator of each block, which determine the factors a, b, and c, which are the elements of A:

$$O = \begin{bmatrix} aB & bB \\ cB & dB \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \otimes B = A \otimes B$$

5. DISCUSSION AND CONCLUSION

5.1. Strengths and Weaknesses of Result

The main strength of the result is the holographic code library, which allowed me -and hopefully others as well- to quickly code holographic pentagon codes. Now adding more features is easier. The weaknesses of the result are that currently there is no way to automatically

generate a holographic pentagon code of n layers. This has to be done manually and is a tedious task that is very prone to error. Another weakness of the result is that besides operator pushing, there are missing tools to study this physical system, such as computing expectation values, or calculating the Ryu-Takayanagi Entropy.

5.2. Outlook

In terms of the code implementation, some possible next steps might be:

- An important next step is to parameterize the number of layers. In [1], the number of pentagons in each layer is given in Appendix C, but it still remains a challenge to come up with a way to algorithmically connect the pentagons.
- Computing the expectation value of a set of operators.
- Compute the Ryu-Takayanagi formula for entropy.

However, to decide what to implement next, I would need guidance to prioritize a direction.

5.3. Open Questions I Have

In terms of physics, I would a deeper understanding of the field, to identify which directions might be promising, but some open questions that come to mind are:

- Are there other quantum error correcting codes in different space-time geometries that act as a AdS/CFT correspondence? If yes, is there a way to derive the error correcting code from a given space-time geometry?
- Could there be a quantum error correcting code for our 3-dimensional space? What is it? How do we get it?
- What intuitions from the Holographic Pentagon Code do we want to gain to better understand quantum gravity in our 3D space?
- Which intuitions are conducent to understand quantum gravity in our 3D space?

Personally I would like to understand the concepts of the Erasure threshold and the Causal Wedge better. I would also like to understand the idea that space-time emerges from entangled states, where the entanglement is protected via quantum error correcting schemes as discussed in [4]. Finally, I would love to keep working on this project and if possible, publish something out of it.

6. ACKNOWLEDGMENTS

I would like to thank Prof. Dr. Geoff Penington for his guidance to recommend this project and for explaining the basics during office hours: thank you!

---

[1] Yoshida B. Harlow D. et al. Pastawski, F., "Holographic quantum error-correcting codes: toy models for the bulk/boundary correspondence," High Energ. Phys. **2015** **149** (2015), https://doi.org/10.1007/JHEP06(2015)149.

[2] "Ads/cft correspondence and quantum error correction," (2022).

[3] Román Orús, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states," Annals of Physics **349**, 117–158 (2014).

[4] Xiao-Liang Qi, "Exact holographic mapping and emergent space-time geometry," (2013).

Appendix A: Resources to learn about Tensor Networks

- Tutorial on Tensor Networks and Quantum Computing with Miles Stoudenmire
- Two really good introductory texts are Hand-waving and Interpretive Dance: An Introductory Course on Tensor Networks and A Practical Introduction to Tensor Networks: Matrix Product States and Projected Entangled Pair States
- Examples code of tensor networks for quantum information can be found in tensors.net and in the TensorNetwork Library documentation.