Space-Time Circuit-to-Hamiltonian Construction Applied to a MERA Circuit

Bachelor Thesis

by

Friederike Metz

Faculty of Mathematics, Computer Science, and Natural Sciences
RWTH Aachen University

July 30, 2015

Supervised by
Prof. Dr. B. M. Terhal
Institute for Quantum Information

Eidesstattliche Erklärung

Ich versichere, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Aachen, den 30. Juli 2015

Friederike Metz

Acknowledgements

First of all, I want to thank professor Barbara Terhal for supervising this thesis, providing me with all the ideas on which it is based, and for helping me whenever I had questions or problems. I would also like to thank Nikolas Breuckmann, who assisted me especially in writing the computer program. Finally, I thank Joel Schumacher and Darius Reinert for their great support throughout the last three months.

Contents

List of Figures

Notation

$\mathbb{I}$	Identity operator
$\|0\rangle^N$	Tensor product of N same states: $\|0\rangle \otimes ... \otimes \|0\rangle$
$\mathbb{V}^N$	Tensor product of N vector spaces: $\mathbb{V} \otimes ... \otimes \mathbb{V}$
n	(Half) the number of input qubits in a circuit
D	Depth of a circuit
z	Scale parameter in the MERA; finest scale corresponds to $z = 0$
$\|\xi\rangle$	Qubit register at initial time
$\|\mathbf{t}\rangle$	Time configuration: Times of all qubits in a tensor product $\|\mathbf{t}\rangle = \|t_1\rangle \otimes ... \otimes \|t_n\rangle$
$\|s\rangle$	String: Representation of time configuration as a string of 0s and 1s
U	Unitary
I	Isometry
$G(V, E)$	Graph G with set of vertices V and edges E
$L(G)$	Laplacian of graph G
$\mathrm{Tr}(\rho)$	Trace of operator ρ
$\|\mathrm{vac}\rangle$	Vacuum state
n^{Tr}	Number of qubits that are traced out
V	Boundary vertex set: Set of vertices, in which a string intersects the boundary of a region
s^V_{min}	Minimal string: Minimally in time evolved string that runs through the vertices of the set V

Outline

This thesis is structured as follows: In the first chapter 1 a brief introduction to quantum computing is given and two models of quantum computation are discussed in detail: The circuit-to-Hamiltonian construction and the space-time circuit-to-Hamiltonian construction. In the second chapter 2 the multi-scale entanglement renormalization ansatz (MERA) is explained and the space-time circuit-to-Hamiltonian construction is applied to a particular MERA circuit. However, a numerical analysis of the circuit Hamiltonian was not successful, since it quickly exceeded the available memory space. Additionally, a second topic was covered within this bachelor project, which was to examine the entanglement entropy of a space-time region in a quantum circuit. This topic is considered in a separate third chapter 3.

1 Circuit Hamiltonians and their Properties

A detailed introduction to quantum computing and quantum information theory is given in reference [13], on which the following paragraphs are based.
Quantum computers are capable of efficiently simulating quantum many-particle systems and thus will have huge implications in condensed matter physics and in all areas that are based on quantum mechanical principles. There are other known applications of quantum computers as well, for example in cryptography.
The commonly used model for quantum computation is the quantum circuit: Classical bits are replaced by qubits, which can be in the states $|0\rangle$, $|1\rangle$, or in any linear combination of the two: $\alpha|0\rangle + \beta|1\rangle$. α and β are complex coefficients and chosen so that the resulting state is normalized to one: $|\alpha|^2 + |\beta|^2 = 1$. Thus, the Hilbert space of a qubit is given by a two dimensional complex vector space $\mathbb{C}^2$. The notion of classical logic gates is replaced by unitary operators, called unitary gates, which act on the qubit states and preserve its length. As a NAND gate is universal for classical computation, it can be shown that two-qubit unitary gates are universal for quantum computation.
There are two constructions in which a quantum circuit is mapped onto a time independent Hamiltonian. In the following sections these two circuit-to-Hamiltonian constructions are introduced. The first one is due to the idea of Feynman [7] and was first used by Kitaev to show that the 5-local Hamiltonian problem is QMA-complete (hard to solve for quantum computers) [10]. The second construction was established by Terhal and Breuckmann in references [2] and [1] and can be interpreted as a generalization of the former one. Both constructions have applications for quantum adiabatic computation and for the complexity class QMA. In the last section of this chapter we consider two realizations of the clock states that are defined within the construction.

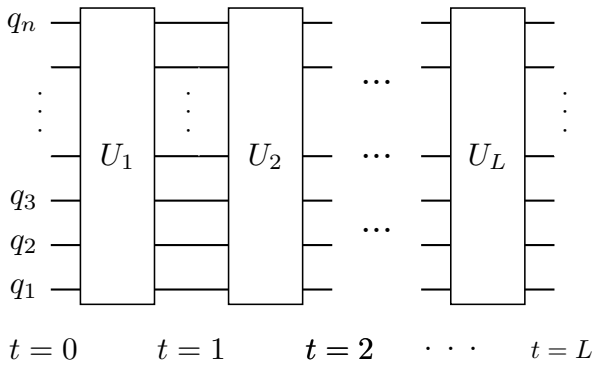

Figure 1.1: A quantum circuit composed of L unitary gates $U_1, ..., U_L$ that act on n qubits $q_1, ..., q_n$. The state of the computation is displayed by one global clock t, which runs from $t = 0$ (initial state) to $t = L$ (output state) in integer steps.

1.1 Circuit-to-Hamiltonian Construction

This section follows reference [10] in a close manner. We introduce a model of quantum computation, in which the quantum circuit is described by the ground state of a Hamiltonian. Suppose a quantum circuit consists of L unitary gates $U_1, ..., U_L$ which act on n qubits $q_1, ..., q_n$ as in figure 1.1. In the circuit-to-Hamiltonian construction one global clock t is defined that takes on values $t = 0, ..., L$, where each of the possible times corresponds to one step in the computation. Hence, initially the clock reads $t = 0$ and when the computation is finished, the time is $t = L$. We associate a state $|t\rangle$ to each time with a Hilbert space $\mathcal{H}_{\text{time}} = \mathbb{C}^{L+1}$. A state $|\Psi\rangle$ of the circuit system is now given by the qubit register $|\xi_t\rangle$ and the time register $|t\rangle$ in a tensor product: $|\Psi\rangle = |\xi_t\rangle \otimes |t\rangle \in \mathbb{C}^{2^n} \otimes \mathbb{C}^{L+1}$. The qubit register depends on the time step t, since each application of a unitary gate changes its internal state.
The Hamiltonian H_{circuit} corresponding to the circuit in figure 1.1 is then defined by [10]:

$$H_{\text{circuit}} = \sum_{t=1}^{L} H_t = \sum_{t=1}^{L} -(\overbrace{U_t \otimes |t\rangle \langle t-1|}^{1} + \overbrace{U_t^\dagger \otimes |t-1\rangle \langle t|}^{2}) + \overbrace{\mathbb{I} \otimes |t\rangle \langle t| + \mathbb{I} \otimes |t-1\rangle \langle t-1|}^{3} \tag{1.1}$$

First of all, notice that for each unitary gate in the circuit, there is one term H_t in the circuit Hamiltonian. Term 1 maps the clock state $|t-1\rangle$ to the state $|t\rangle$ one time step ahead while the corresponding gate U_t of the circuit is applied on the qubit register. Term 2 is the hermitian conjugate of the first one and corresponds to time running backwards. Thus, H_{circuit} becomes a hermitian operator. The last term causes the circuit Hamiltonian to be positive semi-definite so that it has a zero energy ground state. In this case the identity $\mathbb{I}$ acts on the qubit register. In the following equations the identity operators will be omitted. Hence, when no other unitary gate is specified, the identity is supposed to act on the qubit register.

We claim that the ground state of H_{circuit} is given by:

$$|\Psi_{\text{history}}\rangle = \frac{1}{\sqrt{L+1}} \sum_{t=0}^{L} U_t ... U_1 |\xi\rangle \otimes |t\rangle \tag{1.2}$$

It is called the history state of the circuit, since it is the uniform superposition over all computation states. $|\xi\rangle$ denotes the initial state of the qubit register at time $t = 0$. Notice that the ground state is highly degenerate, because $|\xi\rangle$ can be any initial state of the circuit. If we measure the time in the ground state $|\Psi_{\text{history}}\rangle$, we obtain $t = L$ with probability $1/(L+1)$ and can then read off the result of the computation in the qubit register. If L is large and thus the probabilities to measure a specific time is low, we can always add identity gates at the end of the circuit and increase the probability of measuring a time $t \geq L$ in this way.
Let us prove that the history state is the ground state of the circuit Hamiltonian, $H_{\text{circuit}} |\Psi_{\text{history}}\rangle = 0$:

Proof.

$$H_{\text{circuit}} |\Psi_{\text{history}}\rangle = \frac{1}{\sqrt{L+1}} \sum_{t'=1}^{L} H_{t'} \sum_{t=0}^{L} U_t ... U_1 |\xi\rangle \otimes |t\rangle \tag{1.3}$$

From the first sum we consider one arbitrary term, say $H_{t'}$:

$$\begin{aligned}
&H_{t'} \sum_{t=0}^{L} U_t ... U_1 |\xi\rangle \otimes |t\rangle \\
&= \sum_{t=0}^{L} \Big(-U_{t'} \otimes |t'\rangle \langle t'-1| - U_{t'}^{\dagger} \otimes |t'-1\rangle \langle t'| + |t'\rangle \langle t'| + |t'-1\rangle \langle t'-1| \Big) U_t ... U_1 |\xi\rangle \otimes |t\rangle \\
&= -\overbrace{U_{t'} U_{t'-1} ... U_1}^{U_{t'} ... U_1} |\xi\rangle \otimes |t'\rangle - \overbrace{U_{t'}^{\dagger} U_{t'} ... U_1}^{U_{t'-1} ... U_1} |\xi\rangle \otimes |t'-1\rangle + U_{t'} ... U_1 |\xi\rangle \otimes |t'\rangle \\
&\qquad\qquad + U_{t'-1} ... U_1 |\xi\rangle \otimes |t'-1\rangle \\
&= 0
\end{aligned} \tag{1.4}$$

Since t' was chosen arbitrarily, every term in 1.3 gives zero. Therefore:

$$H_{\text{circuit}} |\Psi_{\text{history}}\rangle = 0 \tag{1.5}$$

□

The circuit Hamiltonian has a nice property. We can "rotate away" the gates U_t in its expression (equation 1.1) by a unitary transformation with the operator $W = \sum_{t=0}^{L} U_t ... U_1 \otimes |t\rangle \langle t|$:

$$\begin{aligned}\tilde{H}_{\text{circuit}} = W^\dagger H_{\text{circuit}} W &= \sum_{t=1}^{L} W^\dagger H_t W \\ &= \sum_{t=1}^{L} \left(-|t\rangle\langle t-1| - |t-1\rangle\langle t| + |t\rangle\langle t| + |t-1\rangle\langle t-1|\right) \end{aligned} \tag{1.6}$$

The resulting Hamiltonian $\tilde{H}_{\text{circuit}}$ acts only trivially (with the identity) on the qubit register. It corresponds to the Hamiltonian of a particle hopping on a line whose position is given by t.

Proof. First let us prove that $W = \sum_{t=0}^{L} U_t ... U_1 \otimes |t\rangle\langle t|$ is a unitary operator:

$$\begin{aligned} WW^\dagger &= \sum_{t=0}^{L}\sum_{t'=0}^{L} U_t ... U_1 (U_{t'} ... U_1)^\dagger \otimes |t\rangle\langle t|t'\rangle\langle t'| \\ &= \sum_{t=0}^{L} U_t ... U_1 U_1^\dagger ... U_t^\dagger \otimes |t\rangle\langle t| \\ &= \sum_{t=0}^{L} \mathbb{I} \otimes |t\rangle\langle t| \end{aligned} \tag{1.7}$$

To prove equation 1.6, we again consider an arbitrary term $W^\dagger H_t W$:

$$W^\dagger H_t W = W^\dagger \left(-\underbrace{U_t \otimes |t\rangle\langle t-1|}_{1} - \underbrace{U_t^\dagger \otimes |t-1\rangle\langle t|}_{2} + \underbrace{|t\rangle\langle t| + |t-1\rangle\langle t-1|}_{3} \right) W \tag{1.8}$$

$$\begin{aligned} 1: \quad & \left(\sum_{t_2=0}^{L} (U_{t_2} ... U_1)^\dagger \otimes |t_2\rangle\langle t_2|\right) U_t \otimes |t\rangle\langle t-1| \left(\sum_{t_1=0}^{L} U_{t_1} ... U_1 \otimes |t_1\rangle\langle t_1|\right) \\ &= (U_t ... U_1)^\dagger U_t \left(U_{t-1} ... U_1\right) \otimes |t\rangle\langle t-1| \\ &= \left(U_1^\dagger ... U_t^\dagger\right)\left(U_t ... U_1\right) \otimes |t\rangle\langle t-1| \\ &= \mathbb{I} \otimes |t\rangle\langle t-1| \end{aligned} \tag{1.9}$$

Analog, the second and third term can be calculated:

$$\begin{aligned} 2: \quad &= (U_{t-1} ... U_1)^\dagger U_t^\dagger \left(U_t ... U_1\right) \otimes |t-1\rangle\langle t| \\ &= \left(U_1^\dagger ... U_{t-1}^\dagger\right)\left(U_{t-1} ... U_1\right) \otimes |t-1\rangle\langle t| \\ &= \mathbb{I} \otimes |t-1\rangle\langle t| \end{aligned} \tag{1.10}$$

$$3: \quad = \mathbb{I} \otimes |t\rangle\langle t| + \mathbb{I} \otimes |t-1\rangle\langle t-1| \tag{1.11}$$

□

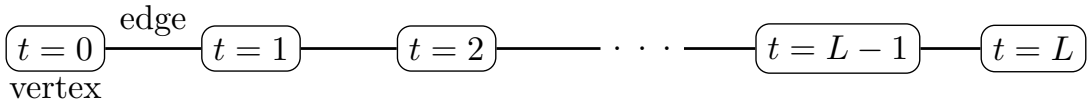

Figure 1.2: The Graph $G = (V, E)$ associated to the circuit Hamiltonian H_{circuit}. Each time step t is one vertex in this graph and the edges connect all times that differ by one computation step. The graph G is simple and connected.

We can associate a simple, undirected graph $G = (V, E)$ with vertex set V and a set of edges E to the circuit Hamiltonian $\tilde{H}_{\text{circuit}}$ (figure 1.2). Each time step is one vertex in this graph and an edge $(t-1, t)$ connects each time step t to the former one $t-1$. A graph can be described by its Laplacian matrix $L(G)$ that is defined in the following way:

Definition 1.1. *Laplacian matrix*
The Laplacian matrix $L(G)_{t,t'}$ of a simple graph $G = (V, E)$ is defined by

$$L(G)_{t,t'} \equiv D(G)_{t,t'} - A(G)_{t,t'} \tag{1.12}$$

$D(G)_{t,t'}$ and $A(G)_{t,t'}$ are the degree and adjacency matrix of the graph G:

$$D(G)_{t,t'} \equiv \begin{cases} \deg(t) & \text{if } t = t' \\ 0 & \text{otherwise} \end{cases} \tag{1.13}$$

$$A(G)_{t,t'} \equiv \begin{cases} 1 & \text{if } (t, t') \in E \\ 0 & \text{otherwise} \end{cases} \tag{1.14}$$

where $\deg(t)$ is the number of edges that determine on vertex t.

The transformed circuit Hamiltonian $\tilde{H}_{\text{circuit}}$ represented in the $\{|t\rangle\}$ basis equals the Laplacian matrix $L(G)_{t,t'}$ of graph G:

$$\langle t|\, \tilde{H}_{\text{circuit}}\, |t'\rangle = L(G)_{t,t'} \begin{pmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & -1 & \ddots & \ddots & \\ & & & \ddots & 2 & -1 \\ & & & & -1 & 1 \end{pmatrix} \tag{1.15}$$

One important result of spectral graph theory, which we use later on, is given by the following proposition.

Proposition 1.2. *The smallest eigenvalue of the Laplacian matrix of a connected graph is zero and the corresponding eigenvector is the uniform superposition over all vertices in this graph [12].*

The graph G of figure 1.2 is connected. Hence, with this proposition we could have also shown that the history state (equation 1.2) is the ground state of the circuit Hamiltonian.

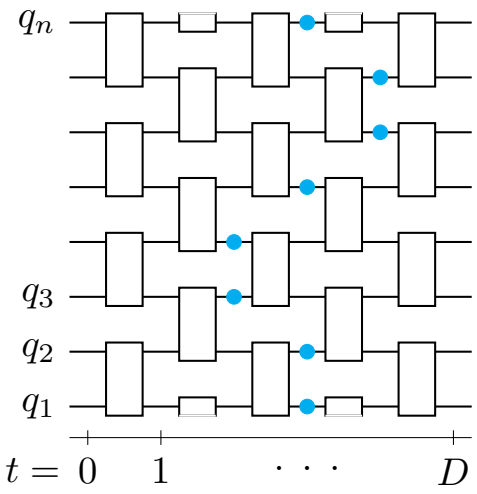

Figure 1.3: The circuit that is used for the space-time circuit-to-Hamiltonian construction consists of two-qubit gates arranged in an alternating manner. The clocks of every qubit do not need to be synchronized anymore. An example of a valid time configuration within the circuit is displayed by the blue dots.

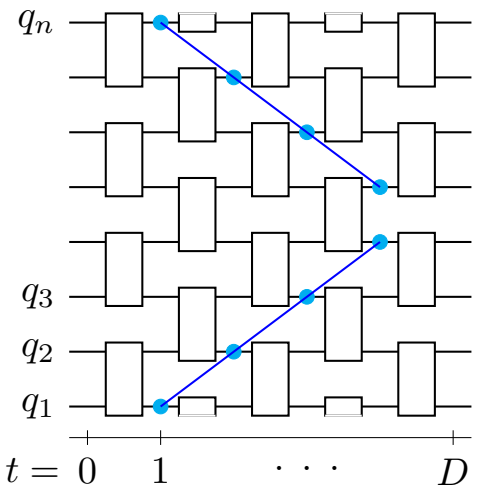

Figure 1.4: An example of a time configuration which displays the past causal cone (blue lines) of the two rightmost qubits. The qubits lying on the diagonal lines are light-like separated and thus cannot influence each other. Only qubits which are time-like separated can affect each others internal state.

1.2 Space-Time Circuit-to-Hamiltonian Construction

The space-time circuit-to-Hamiltonian construction was worked out by Breuckmann and Terhal in references [1] and [2], on which this section is also based. It was motivated by the two dimensional interacting fermion model of reference [11]. Similarly to the former circuit-to-Hamiltonian construction, this one can be used for universal adiabatic quantum computation [8] and for QMA completeness proofs [2].

As a major difference between the two constructions, every qubit q_i now gets its own clock $|t_i\rangle$ and the times of the qubits are not synchronized in general. In the following we allow only nearest neighbor interactions within the circuit. Thus, it consists exclusively of two-qubit gates as in figure 1.3. However, two-qubit gates are universal for quantum computation, so this represents no restriction on the number of possible computations. The unitary gates are arranged in an alternating manner with the total number of time steps given by the depth D of the circuit. Hence, the time of each qubit can range between $t = 0, ..., D$. The time register is now given by a tensor product of time states of all qubits, called time configuration: $|\mathbf{t}\rangle = |t_1\rangle \otimes ... \otimes |t_n\rangle = |t_1, ..., t_n\rangle$. The time configuration, in which each time in the tensor product equals zero, denotes the initial computation state. In this construction the Hilbert space associated to the time configurations is $\mathcal{H}_{\text{time}} = \mathbb{C}^{(D+1)^n}$.

However, not every state $|\mathbf{t}\rangle \in \mathcal{H}_{\text{time}}$ corresponds to a possible time configuration within the circuit. When a unitary gate is applied on two qubits, the time register of both qubits is updated

and thus both hop together one time step ahead. Hence, one qubit cannot be past a gate while its respective partner qubit is not. Possible time configurations are called valid, while impossible ones are called invalid. For two partner qubits q_i, q_j, on which the unitary gate $U_t(q_i, q_j)$ acts at time t, an invalid time configuration would imply either $t_i < t \wedge t_j \geq t$ or $t_j < t \wedge t_i \geq t$. Figures 1.3 and 1.4 both display an example of valid time configurations.
Let us observe figure 1.4 in more detail. The region, bordered by the two blue lines, displays the backwards causal cone of the two rightmost qubits. All qubits that lie within this causal cone can possibly influence the two qubits on the right, while all qubits that lie outside of the causal cone cannot have an affect on them. In this sense, the qubits inside the causal cone are time-like separated, and the qubits outside of it are space-like separated. Qubits, which lie on diagonal lines as in figure 1.4, are then light-like separated and these kind of time configurations cannot evolve forward. Using this notation, an invalid time configuration occurs, if there are at least two qubits that are time-like separated.
The Hamiltonian of the former circuit-to-Hamiltonian construction (equation 1.1) has to be changed slightly to account for the multiple time states. For each unitary gate $U_t(q_i, q_j)$ in the circuit that acts at time t on the qubits q_i, q_j, one term of the following form is introduced in the circuit Hamiltonian [2]:

$$\begin{aligned} H_t(q_i, q_j) = &- \left(U_t(q_i, q_j) \otimes |t_i, t_j\rangle \langle t - 1_i, t - 1_j| + h.c.\right) \\ &+ |t_i, t_j\rangle \langle t_i, t_j| + |t - 1_i, t - 1_j\rangle \langle t - 1_i, t - 1_j| \end{aligned} \tag{1.16}$$

Thus, H_{circuit} is the sum over all $H_t(q_i, q_j)$. The first term takes the state where qubits q_i and q_j are both at time $t - 1$ and updates their times to t, while the unitary gate $U_t(q_i, q_j)$ is applied on both internal qubit states. All other qubits are unchanged (the identity acts on them). The following three terms make H_{circuit} hermitian and positive semi-definite as in section 1.1. The circuit Hamiltonian preserves the subspace of valid time configurations and hence in the following we consider H_{circuit} in this subspace.
Analog to section 1.1, we can transform away the gates $U_t(q_i, q_j)$ in the expression of H_{circuit} with the unitary operator $W = \sum_{\text{valid } \mathbf{t}} V(0 \to \mathbf{t}) \otimes |\mathbf{t}\rangle \langle \mathbf{t}|$. In this case $V(0 \to \mathbf{t})$ denotes the product of all gates that need to be applied on the qubit register to reach time configuration $|\mathbf{t}\rangle$ starting from the initial one. The proof works just as for the circuit-to-Hamiltonian construction.
The resulting Hamiltonian $W^\dagger H_{\text{circuit}} W$ acts trivially on the qubit register and can again be associated with a graph $G = (V, E)$ (do not confuse the vertex set V with the operator $V(0 \to \mathbf{t})$). Each valid time configuration $\mathbf{t}$ displays one vertex in the graph G and two vertices $\mathbf{t}, \mathbf{t}'$ are connected by an edge, if there exists a term in the Hamiltonian that can map these time configurations onto each other. As an example, figure 1.5 shows the graph (right) corresponding to the circuit (left) with input qubits $n = 6$ and depth $D = 2$. The circuit Hamiltonian represented in the basis of valid time configurations equals the Laplacian matrix $L_{\mathbf{t},\mathbf{t}'}(G)$ of the obtained graph G:

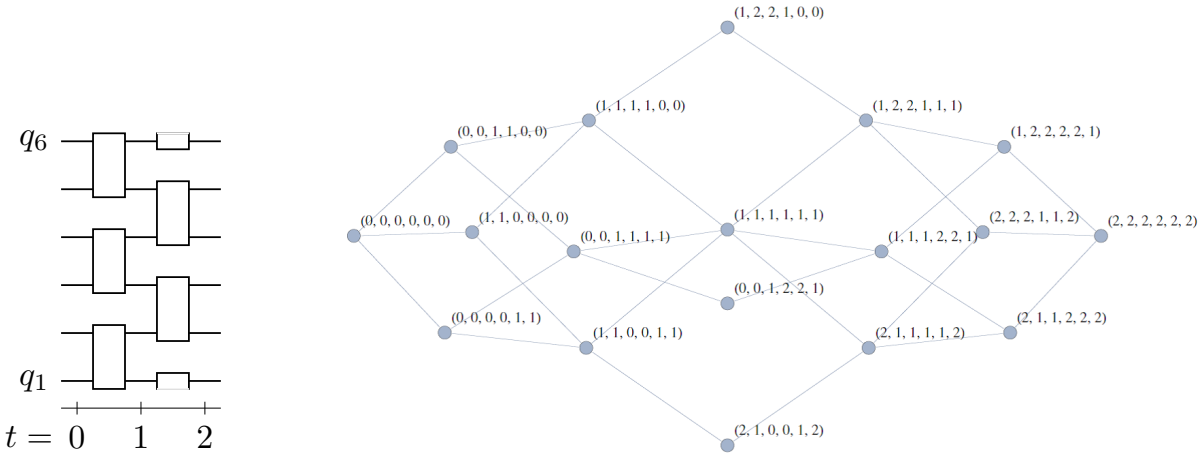

Figure 1.5: (Taken from [1]). The graph G (right) corresponding to the circuit (left) with $n = 6$ and $D = 2$. Each valid time configuration gives rise to one vertex (blue dot) in the graph and an edge (blue line) between two time configurations exists, if H_{circuit} is able to map these time configurations onto each other.

$$\langle \mathbf{t} | \, H_{\text{circuit}} \, | \mathbf{t}' \rangle = L_{\mathbf{t},\mathbf{t}'}(G) \tag{1.17}$$

By applying the circuit Hamiltonian a sufficient number of times to the initial state $|t_1 = 0\rangle \otimes ... \otimes |t_n = 0\rangle$, all valid time configurations can be generated. Thus, the graph G is connected and we can use proposition 1.2, saying that the zero energy ground state of the Laplacian matrix is the uniform superposition over all vertices (valid time configurations):

$$|\tilde{\Psi}_{\text{ground}}\rangle = \frac{1}{\sqrt{N}} \sum_{\text{valid } \mathbf{t}} |\mathbf{t}\rangle \tag{1.18}$$

where N is the total number of valid time configurations.
Transforming this state back by the operator W, we obtain the history state as the ground state of the circuit Hamiltonian:

$$|\Psi_{\text{history}}\rangle = \frac{1}{\sqrt{N}} \sum_{\text{valid } \mathbf{t}} V(0 \to \mathbf{t}) \, |\xi\rangle \otimes |\mathbf{t}\rangle \tag{1.19}$$

We computed the ground state of H_{circuit} in the subspace of only valid time configurations. If we do not restrict the Hilbert space to this subspace, we need to add a constraining term to the circuit Hamiltonian that penalizes each invalid time configuration. Only then the ground state is still given by the superposition of just valid configurations. In the next section we examine two possible representations of the time states and consider such penalties.

1.3 Clock Representations

There are several ways how the time state $|t\rangle$ of a qubit can be represented. In the following two different representations are introduced: the pulse clock and the string.

1.3.1 Pulse Clock

This section is based on reference [1].
In the pulse clock representation the time state $|t\rangle$ is given by $D+1$ qubits from which D qubits are in the state $|0\rangle$ and exactly one qubit is in the state $|1\rangle$ that points to the time. These qubits are called time or clock qubits, so we do not confuse them with the actual qubits on which the computation is done. The different time states are then defined as follows:

$$\begin{aligned} t \in \{0, ..., D\} \\ |t=0\rangle &= |\underbrace{1000...0}_{\text{length } D+1}\rangle \\ |t=k\rangle &= |0...0\underbrace{1}_{k+1}0...0\rangle \\ |t=D\rangle &= |0...001\rangle \end{aligned} \tag{1.20}$$

In the space-time circuit-to-Hamiltonian construction each qubit has its own clock and thus each of their time states is represented by one pulse clock. For example, if qubit q_i is at time step $t=3$, the time configuration would read $|\mathbf{t}\rangle = |t_1\rangle \otimes ... \otimes |t_{i-1}\rangle \otimes |000100\rangle_i \otimes ... \otimes |t_n\rangle$.
Using the pulse clock representation we can rewrite the time transition operators in the circuit Hamiltonian (equation 1.16):

$$\begin{aligned} H_t(q_i, q_j) = &- (U_t(q_i, q_j) \otimes |01\rangle\langle 10|_{i;t,t+1} \otimes |01\rangle\langle 10|_{j;t,t+1} + h.c.) \\ &+ |1\rangle\langle 1|_{i;t+1} \otimes |1\rangle\langle 1|_{j;t+1} + |1\rangle\langle 1|_{i;t} \otimes |1\rangle\langle 1|_{j;t} \end{aligned} \tag{1.21}$$

The notation $|1\rangle\langle 1|_{i;t+1}$ denotes that the time state of qubit q_i reads a 1 at position $t+1$ implying that qubit q_i is located at time step t. The Hilbert space of the time states on which the circuit Hamiltonian acts is now given by $\mathbb{C}^{2^{(D+1)^n}}$. It is substantially larger than it was before ($\mathbb{C}^{(D+1)^n}$). Therefore, besides penalizing all invalid time configurations, we also need to introduce a constraining term H_{legal}, which penalizes all time states that do not correspond to a time. These are those states that involve more or less than one clock qubit in the state $|1\rangle$:

$$H_{\text{legal}} = \sum_i \left(|0...0\rangle\langle 0...0|_i + \sum_{\substack{t,t'=0 \\ t\neq t'}}^{D} |1\rangle\langle 1|_{i;t+1} \otimes |1\rangle\langle 1|_{i;t'+1} \right) \tag{1.22}$$

If any qubit q_i has a corresponding time state that is illegal, H_{legal} will rise its energy and thus it is not part of the zero energy ground state. The fist term in equation 1.22 penalizes all time states without a pointer and the second term penalizes all states, which contain at least two 1s. We are also able to penalize all states that correspond to invalid time configurations as defined in section 1.2. The term in the Hamiltonian that forces the ground state to be a linear combination of only valid time configurations is the following:

$$\begin{aligned} H_{\text{caus}} = &\sum_{\substack{i+t \\ \text{even}}} |1\rangle\langle 1|_{i;t+1} \left(\mathbb{I} - |1\rangle\langle 1|_{i+1;t+2} - |1\rangle\langle 1|_{i+1;t+1}\right) \\ &+ \sum_{\substack{i+t \\ \text{odd}}} |1\rangle\langle 1|_{i;t+1} \left(\mathbb{I} - |1\rangle\langle 1|_{i+1;t} - |1\rangle\langle 1|_{i+1;t+1}\right) \end{aligned} \tag{1.23}$$

If the clock of qubit q_i reads time t and $t+i$ is even (odd), its adjacent qubit q_{i+1} has to be either at the same time or one time step ahead (before), otherwise the configuration is invalid. Notice that the Hamiltonian of equation 1.21 is a 6-local operator in this representation. It acts 2-locally on the qubit register and each of the time transition operators, which involves two adjacent clock qubits, is now 2-local as well. However, H_{legal} is clearly non-local. Hence, the pulse clock is not useful for applications in QMA. If a local Hamiltonian with local constraining terms is needed, a better representation one might want to choose is the domain wall clock. In this case the time is given by the position of a domain wall 10 in a series of 1s followed by a series of 0s, for example: $|t=3\rangle = |1110000\rangle$. In this way the time transition operators in the circuit Hamiltonian can be built 3-locally and the constraining terms become local as well.

1.3.2 String

The second representation of the time states we introduce are called strings. This representation is especially useful for quantum adiabatic computation as shown in [8]. The following section is built on reference [8].

In order to define a string, we visualize the circuit of section 1.2 in a different way as displayed in figure 1.6. The circuit is now given by a $n \times n$ grid, in which each square represents one unitary gate and each edge can be occupied by one qubit. Qubits can hop horizontally from edge to edge with time increasing from left to right. A major difference to the former circuit is that not all qubits start at time $t = 0$ and end at time $t = D$. For example qubit q_3 is initially at time step $t = 2$ and does not have a time before that. Notice that the total number of qubits in this circuit is given by $2n$.

Figure 1.7 shows two possible time configurations, drawn as a purple and a blue line. Each valid time configuration can be represented by a series of 0s and 1s in the following way: Starting at the lowermost vertex, if qubit q_1 is located on the left edge, we note a 0, if it lies on the right edge, we note a 1. We follow the line to the next vertex in the grid and decide again if qubit q_2 resides on the left (0) or right edge (1). We continue this procedure until the uppermost vertex

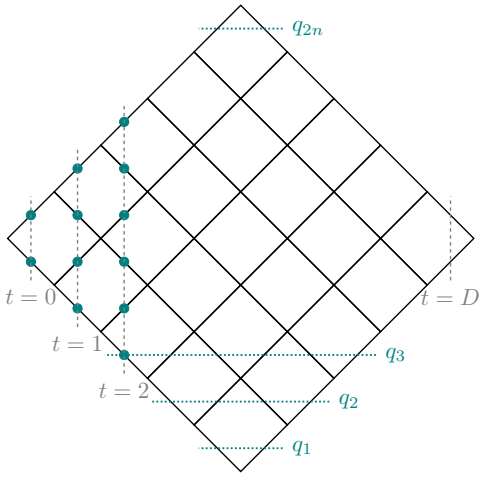

Figure 1.6: The unitary circuit represented as a grid. Each square corresponds to a unitary gate and each edge can be occupied by one qubit as long as the resulting time configuration is valid. When a unitary gate is applied on two qubits, they hop one edge two the right. In this circuit qubits start and end at different time steps opposed to the circuit discussed before.

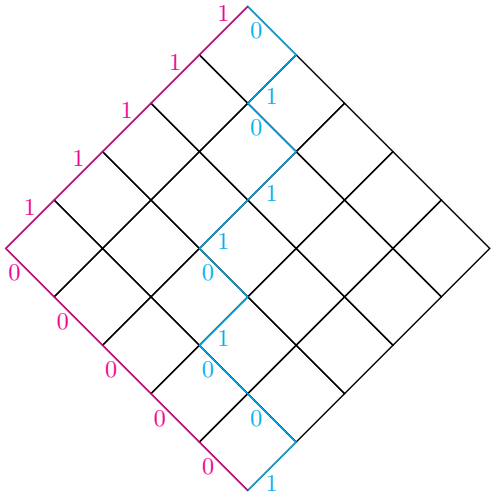

Figure 1.7: A valid time configuration can be drawn as a line of qubits and is represented as a series of 0s and 1s. The time configuration corresponding to the blue line is given by $|\mathbf{t}\rangle \equiv |s\rangle = |1001011010\rangle$. The state associated to the purple line is defined as the initial string $|s_{\text{init}}\rangle = |0...01...1\rangle$.

and are left with a unique state, called string $|s\rangle$, given by the obtained series of 0s and 1s. For example the time configuration given by the blue line is: $|\mathbf{t}\rangle \equiv |s\rangle = |1001011010\rangle$. The string corresponding to the purple line is defined as the initial string: $|s_{\text{init}}\rangle = |0...01...1\rangle$.
Let us examine how a string is changed when a unitary gate is applied and the corresponding qubits move one time step further. The string of qubits always has to be a connected line in order to display a valid time configuration. Thus, the string segments that read 01 are the only ones that can propagate forward. After the application of the unitary gate, the string segment 01 is mapped to 10. For backwards propagation it is reversed: $10 \rightarrow 01$. Therefore, using the string representation each term in the circuit Hamiltonian can be compactly written as:

$$H_t(q_i, q_j) = -(U_t(q_i, q_{i+1}) \otimes |10\rangle\langle 01|_{i,i+1} + h.c.) + |01\rangle\langle 01|_{i,i+1} + |10\rangle\langle 10|_{i,i+1} \tag{1.24}$$

Only those strings with Hamming weight equal to n (number of 1s equals number of 0s), correspond to a possible string configuration. Thus, we again need to add a constraining term to the circuit Hamiltonian that penalizes all strings with Hamming weight unequal n.
Within the subspace of valid strings, the action of the circuit Hamiltonian on the string states

equals a slightly modified Heisenberg model with Hamiltonian $\tilde{H}_{XYZ-\mathbb{I}}$ (X, Y, Z denote the Pauli matrices):

$$\tilde{H}_{XYZ-\mathbb{I}} = -\frac{1}{2}\sum_{i=1}^{2n-1}\Big(\underbrace{X_iX_{i+1}+Y_iY_{i+1}}_{2(|10\rangle\langle 01|_{i,i+1}+|01\rangle\langle 10|_{i,i+1})} + \underbrace{Z_iZ_{i+1}-\mathbb{I}_i\mathbb{I}_{i+1}}_{-2(|01\rangle\langle 01|_{i,i+1}+|10\rangle\langle 10|_{i,i+1})} \Big) \tag{1.25}$$

Hence, the circuit Hamiltonian can be analyzed in terms of the widely understood Heisenberg model.

Furthermore, we could examine the time evolution of the strings with the propagating Hamiltonian H_{prob}, starting from the initial string $|s_{\text{init}}\rangle = |0...01...1\rangle$. H_{prob} is given by the adjacency matrix, which are only the first two, off-diagonal terms in equation 1.24. In reference [8] the dynamics of the strings are studied in the limit of infinite n, using the quantum walk on Young's lattice. Each valid string can be uniquely mapped to one Young diagram, since each diagram can be interpreted as displaying the gates (squares) lying left to a string in figure 1.7. Using the exactly computable quantum walk on Young's lattice, it is shown that for large times the strings propagate forward as a wavefront with constant velocity.

2 Multi-Scale Entanglement Renormalization Ansatz (MERA)

The MERA is a tensor network that can efficiently approximate ground states of quantum critical systems [5]. Besides the wide applications for quantum many-particle physics, the MERA seems to be closely related to a theory of quantum gravity: For the conformal field theory (CFT) of a quantum critical spin chain, the holographic geometry of the MERA describes an anti-de Sitter (AdS) space and thus realizes an AdS/CFT correspondence [14].
The next section of this chapter introduces the MERA as a tensor network. In the second section we apply the space-time circuit-to-Hamiltonian Construction to a MERA circuit and analyze the associated Laplacian matrix and its spectrum. The main questions are to examine the properties of the Laplacian (eigenvalue gap, symmetries, ...), and analyze the forward propagation of the strings. However, we were not able to answer these questions within this bachelor project. The chapter is concluded with an outlook on further research.

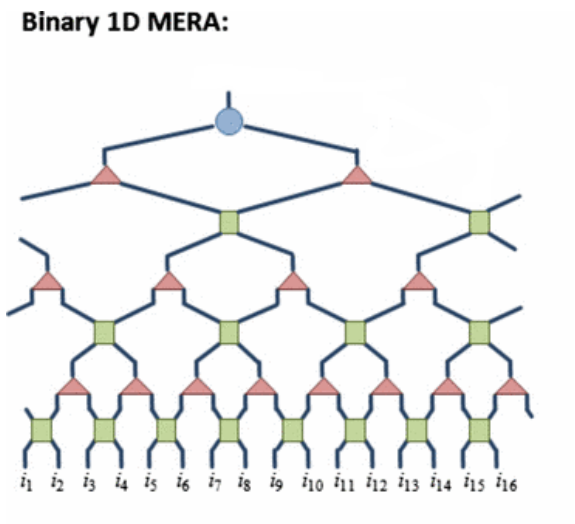

Figure 2.1: (Taken from [3]). The binary one dimensional MERA describing a lattice composed of 16 sites with periodic boundary condition. Each green square represents a unitary, each red triangle represents an isometry. Unitaries disentangle the lattice sites while isometries induce a coarse graining transformation. The second (vertical) dimension corresponds to different length scales in the system and describes a holographic geometry.

2.1 Tensor Networks and the MERA

The following introduction to tensor networks and the MERA is obtained from references [4] and [5]. Tensor networks can describe ground states of local Hamiltonians on a lattice by a set of tensors that are contracted according to a specific geometry. Different geometries of the underlying lattice give rise to different tensor networks. The most common ones are matrix product states (MPS), which approximate ground states on a one dimensional lattice, projected entangled pair states (PEPS) for higher dimensional lattices, and the multi-scale entanglement renormalization ansatz (MERA). The latter serves as an efficient description of quantum critical systems in arbitrary dimensions.

Let us understand the MERA, which was first proposed by Vidal in [18]. Suppose we want to describe a one dimensional lattice L composed of N sites. The sites of the lattice could represent any physical system like spins, qubits, or atoms. The Hilbert space assigned to each site is denoted by $\mathbb{V}$. Thus, the Hilbert space of the whole lattice is given by $\mathbb{V}_L = \mathbb{V}^N$. Figure 2.1 displays a binary MERA describing a lattice of 16 sites $i_1, ..., i_{16}$ with periodic boundary condition. It involves two different kind of tensors, called isometries I (red triangles) and unitaries U (green squares). These tensors are arranged in alternating layers as in figure 2.1. Two successive layers of unitaries and isometries correspond to a coarse-graining transformation, which maps the initial lattice L to a coarser lattice L' that contains only $N/2$ sites with Hilbert space $\mathbb{V}'^{N/2}$. Unitaries map two sites of the lattice again on two sites, hence build a (2,2) tensor, while isometries map one site of the coarser lattice onto two sites of the finer one and thus describe a (1,2) tensor:

$$U: \quad \mathbb{V}^2 \to \mathbb{V}^2 \tag{2.1}$$

$$I: \quad \mathbb{V}' \to \mathbb{V}^2 \tag{2.2}$$

The unitaries are often referred to as disentanglers, because they remove the entanglement between lattice sites.

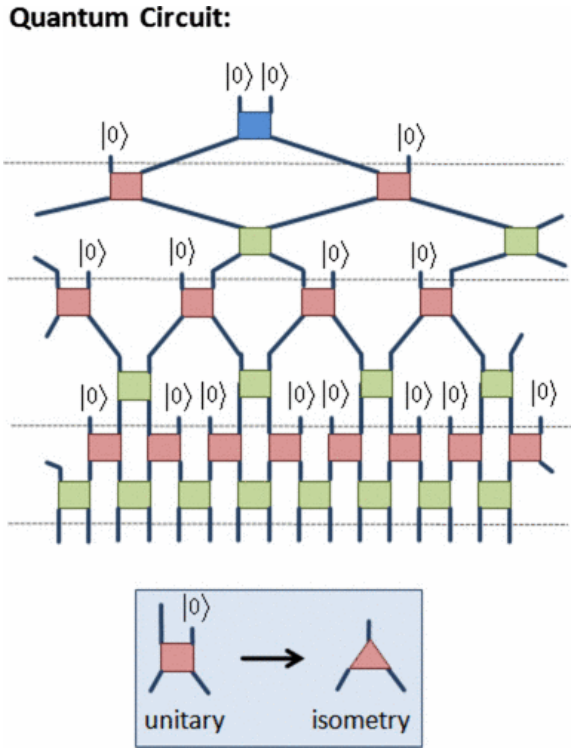

Figure 2.2: (Taken from [3]). Each isometry can be interpreted as a unitary gate when fixing a second input qubit in the state $|0\rangle$. In this realization the input state of the circuit is given by N ancilla qubits: $|0\rangle^N$. The output of the computation corresponds to the lowermost wires.

The tensors in the MERA have one important property, which is indirectly implied by their names:

$$U^\dagger U = \mathbb{I}^2 \qquad\qquad I^\dagger I = \mathbb{I}' \tag{2.3}$$

There exists a major difference between the MERA and other tensor networks like the MPS and PEPS: A MERA describing a d-dimensional lattice always involves an extra dimension, which corresponds to different length scales of the system. Thus, the MERA is able to effectively describe the properties of the system at different scales. Furthermore, the geometry generated by the MERA does not reflect the actual physical geometry of the lattice, but a holographic one given by the entanglement in the wave function of the system. The coarse graining transformations in the MERA from smaller scales to larger ones are therefore known as entanglement renormalization.
In the following sections we interpret the MERA as a quantum circuit. Hence, each site represents a qubit with a Hilbert space given by $\mathbb{V} = \mathbb{C}^2$. Isometric tensors can be understood as unitary gates since we can always add an extra ancilla qubit (initialized in the state $|0\rangle$) to the input. Figure 2.2 illustrates this procedure. In this way the quantum circuit can be seen to have the input state $|0\rangle^N$ and an output state $|\xi\rangle$, which is obtained by the successive application of the gates on the initial state. Notice that the ancilla qubits are "mixed in" at different computation steps throughout the circuit. [3]

2.2 Circuit Hamiltonian

The space-time circuit-to-Hamiltonian construction will be applied to a MERA circuit. This particular MERA is similar to the binary MERA discussed in the section before and visualized in figure 2.3. Each square represents a unitary gate and each six-sided figure represents an isometric gate. Edges can be occupied by a qubit, but again only time configurations that correspond to a valid computation step are allowed. In our MERA isometries take 2 qubits as their input and have 4 output qubits (or equivalently isometries have 4 input qubits from which 2 are ancilla qubits) opposed to the binary MERA where an isometry has only 1 input and 2 output qubits. However, the result is the same: After every second time step the number of qubits is roughly doubled, so for the ease of visualization we work with our MERA of figure 2.3. It can be characterized by two parameters. One being the number of input qubits $2n$ of the whole circuit and the second being the final time step D at which the computation is finished. Notice that some qubits only exist at later time steps. This fact makes a numbering of all the qubits complicated. To avoid it, we again represent a valid time configuration as a string s (series of 0s and 1s) similarly to section 1.3.2.
Starting from the lowest vertex, if a qubit is located on the left edge, we note a 0, if it lies on the right edge, we note a 1. Continuing the line of qubits until the uppermost vertex, we end up with a string that uniquely defines the corresponding state of the computation (if all unitaries and isometries left to the string are known). An example of such a string is shown in figure 2.4. In contrast to the circuit analyzed in section 1.3.2, two arbitrary strings may have different lengths, e.g. two strings differing only by an application of an isometry have lengths that differ by 2. However, a string always involves the same number of 0s and 1s, because it is fixed to start and end at the top and bottom vertices of the MERA.
The initial string is defined as $|s_{\text{init}}\rangle = |0...01...1\rangle$ with n 0s followed by n 1s (see figure 2.4). Let us establish the rule by which all other possible strings can be obtained from the initial string. The rule from section 1.3.2 - when the sequence 01 occurs in the string s, this sequence is mapped to 10 - is not true in this case. The application of an isometry would map the sequence 01 to 1010 increasing the size of the string by 2. For the following rule I assume that the MERA starts with an isometry in the left corner and that it involves an even number of unitaries and isometries at the edges, implying that n is always even.

Proposition 2.1. *Generating all valid strings of a MERA circuit*
Apply the following rule to each obtained string, starting from the initial string $s_{init} = 0...01...1$*: If the 0 of a string segment 01 occurs at an even position in the string, map this string segment to 1010, if it occurs at an odd position, map it to 10.*

Proof. The whole MERA can be built up by periodically continuing the pattern of an isometry followed by three unitaries in parallel (figure 2.5 left). So it suffices to proof the proposition for a circuit consisting of just one isometry (and three following unitaries). If one starts with the string 0011, an isometry is applied leaving the system in the state with string 010101. From this

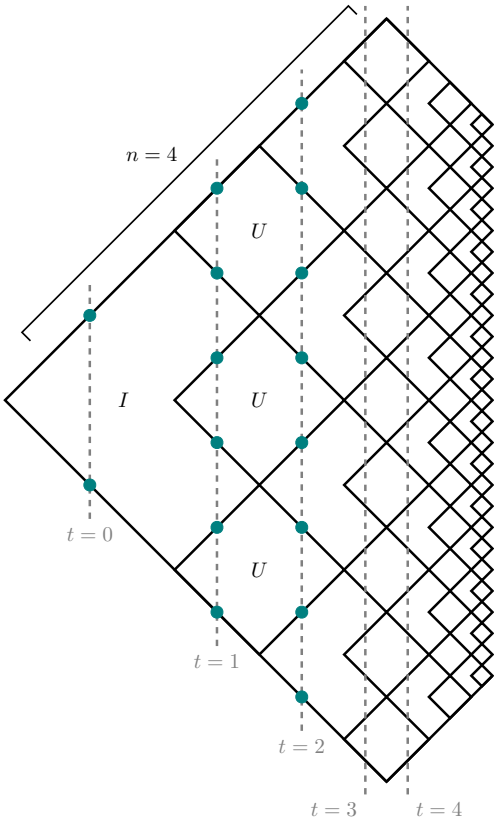

Figure 2.3: A MERA of size $n = 4$ (half the number of input qubits) and $D = 8$ (last time step). Each square corresponds to one unitary (denoted by U), each six-sided object to an isometry (I). Qubits, drawn as dots, can reside on each edge as long as their time configuration is valid. The dashed vertical lines refer to different time steps.

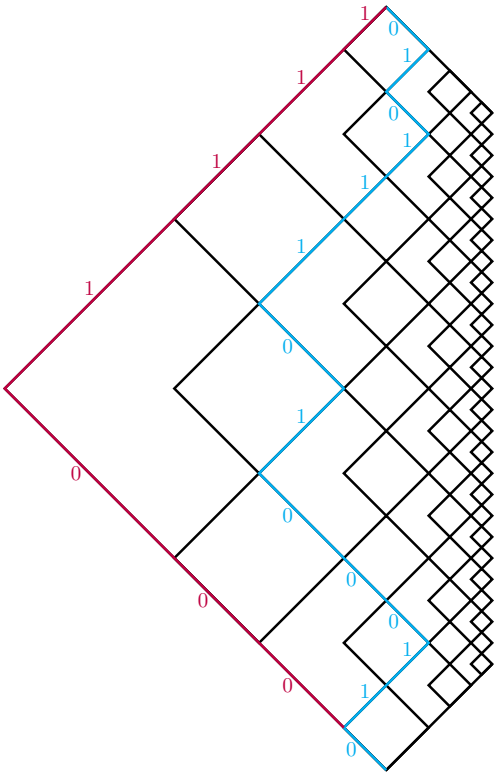

Figure 2.4: Two time configurations (strings) in a MERA circuit. The purple string denotes the initial string, in this case: $s_{\text{init}} = 00001111$. The blue string is one arbitrary valid string: $s = 01100010111010$. Applying an isometry maps the string segment 01 to 1010; applying a unitary maps 01 to 10.

state, one can apply three different unitaries independently, getting one of the following strings: 100101, 011001, 010110, 101001, 100110, 011010, 101010. By applying the rule multiple times on the initial string, we obtain exactly the same states, which proves our conjecture. □

With this rule we are able to generate all possible strings of an arbitrary sized MERA and can compute the associated graph $G = (V, E)$. Just as in section 1.1 each valid string represents one vertex in this graph and two strings s, s' are connected by an edge, if it is possible to map one of these strings to the other by one application of the rule 2.1. Or in other words: $(s, s') \in E$, if there exists either a unitary $U(s \to s')$ or an isometry $I(s \to s')$. As an example figure 2.5 displays the graph corresponding to a MERA with $n = 2$ and $D = 2$. To analyze larger sized MERAs as well, the rule was implemented in a python script (appendix). We will come to its

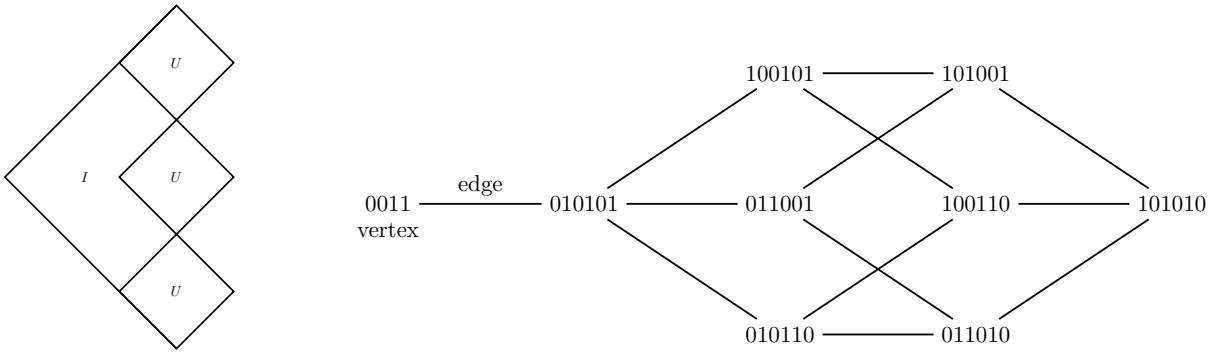

Figure 2.5: The graph (right) corresponding to a $n = 2, D = 3$ MERA (left). Each string becomes one vertex of the graph. Two strings are connected by an edge, if there exists either a unitary or an isometry that maps one string on the other.

discussion in the next section, but first let us work out the Hamiltonian of a MERA circuit. The Hamiltonian associated to the graph G can directly be written down:

$$\tilde{H}^{\text{MERA}}_{\text{circuit}} = \sum_{(s,s')\in E} H(s,s') = \sum_{(s,s')\in E} \left(-|s'\rangle\langle s| - |s\rangle\langle s'| + |s\rangle\langle s| + |s'\rangle\langle s'|\right) \tag{2.4}$$

Represented in the basis of valid strings $\{|s\rangle\}$, $\tilde{H}^{\text{MERA}}_{\text{circuit}}$ is equal to the Laplacian matrix of graph G (definition 1.1). The first two terms of $H(s,s')$ build the antisymmetric adjacency matrix and the last two terms the diagonal degree matrix.

We claim that the actual circuit Hamiltonian, which acts non trivially on the data register, is obtained by a unitary transformation with $W = \sum_{s\in V} V^\dagger(s_{init} \to s) \otimes |s\rangle\langle s|$. The product of all unitaries and isometries needed to reach string s from the initial string s_{init} is again denoted by $V(s_{init} \to s)$ (do not confuse it with the set of vertices V). We prove it similarly to section 1.1.

Proof. Take one arbitrary term of the circuit Hamiltonian, say $H(s,s')$ and transform it by W:

$$W^\dagger H(s,s') W = W^\dagger \left(-\underbrace{|s'\rangle\langle s|}_{1} - \underbrace{|s\rangle\langle s'|}_{2} + \underbrace{|s\rangle\langle s| + |s'\rangle\langle s'|}_{3} \right) W \tag{2.5}$$

$$\begin{aligned}
1: \quad & \left(\sum_{s_2\in V} V(s_{init}\to s_2)\otimes|s_2\rangle\langle s_2|\right)|s'\rangle\langle s|\left(\sum_{s_1\in V} V^\dagger(s_{init}\to s_1)\otimes|s_1\rangle\langle s_1|\right) \\
&= \underbrace{V(s_{init}\to s')}_{V(s\to s')V(s_{init}\to s)}\ V^\dagger(s_{init}\to s)\otimes|s'\rangle\langle s| \\
&= V(s\to s')\otimes|s'\rangle\langle s|
\end{aligned} \tag{2.6}$$

$V(s \to s')$ can either be a unitary or an isometry depending on the strings s and s', namely whether the sequence 01 occurs at an odd or even position in the string s.

$$\begin{aligned} 2: \quad & \left(\sum_{s_2 \in V} V(s_{init} \to s_2) \otimes |s_2\rangle \langle s_2| \right) |s\rangle \langle s'| \left(\sum_{s_1 \in V} V^\dagger(s_{init} \to s_1) \otimes |s_1\rangle \langle s_1| \right) \\ & = V(s_{init} \to s) \underbrace{V^\dagger(s_{init} \to s')}_{(V(s \to s')V(s_{init} \to s))^\dagger} \otimes |s\rangle \langle s'| \\ & = V^\dagger(s \to s') \otimes |s\rangle \langle s'| \end{aligned} \tag{2.7}$$

$$\begin{aligned} 3: \quad & = V(s_{init} \to s)V^\dagger(s_{init} \to s) \otimes |s\rangle \langle s| + V(s_{init} \to s')V^\dagger(s_{init} \to s') \otimes |s'\rangle \langle s'| \\ & = \mathbb{I} \otimes |s\rangle \langle s| + \mathbb{I} \otimes |s'\rangle \langle s'| \end{aligned} \tag{2.8}$$

Since $H(s, s')$ was arbitrary, we obtain the same result for all terms of $\tilde{H}^{\text{MERA}}_{\text{circuit}}$ and therefore can write:

$$H^{\text{MERA}}_{\text{circuit}} = \sum_{(s,s') \in E} \left(-V(s \to s') \otimes |s'\rangle \langle s| - h.c. + |s\rangle \langle s| + |s'\rangle \langle s'| \right) \tag{2.9}$$

□

Because we build the graph G by rule 2.1, we know that it is connected. Therefore, by proposition 1.2 the zero energy ground state of $H^{\text{MERA}}_{\text{circuit}}$ is the history state of the circuit:

$$|\Psi_{\text{history}}\rangle = \frac{1}{\sqrt{|V|}} \sum_{s \in V} V(s_{init} \to s) |\xi\rangle \otimes |s\rangle \tag{2.10}$$

$|V|$ denotes the number of valid strings, hence the size of the vertex set V (again not to confuse with the operator $V(s_{init} \to s)$).

2.2.1 Numerical Analysis

The python script, on which the following discussion is based, can be examined in the appendix. As an input it takes the size of the MERA, the input qubits n and time steps D. First, by rule 2.1, it generates a list of valid strings, referred to as vertex set V. During the computation of the basis states, all nonzero entries in the adjacency matrix, which are characterized by a column and a row index, are memorized and later assembled in a sparse matrix. From the adjacency matrix the degree matrix can be easily calculated: Summing over each row gives twice the degree of the corresponding vertex. The Laplacian is then simply the difference and its eigenvalues and eigenvectors are computed numerically.
As we will see soon the number of valid strings and with it the size of the adjacency matrix get

extremely large and quickly exceed the memory capacity. Hence, it is possible to calculate the set of basis states for MERAs up to size $n = 4, D = 6$ and the Laplacian matrix up to a MERA size of $n = 4, D = 4$. So we are not able to analyze the MERA in figure 2.3. The reason is the doubly exponential growth of the number of valid string states with time t. First, for a string of length l, there are 2^l different possibilities (of course only those with Hamming weight $l/2$ are allowed) and second, after each two time steps the length of a string is roughly doubled due to the isometries. So the main challenges are the exceeding of memory space and long computing times. Therefore, the following steps concerning the program implementation were taken:

- Strings can be interpreted as a binary number. In every case this binary number is converted to an unsigned integer and saved as one. → reduction of memory space
- A dictionary is used to efficiently search in the list of valid strings for the occurrence of each new computed string. If a string is already noted, its position in the list can be directly read off the key of the dictionary (the position is later needed to build the adjacency matrix). Searching in a list would be $\mathcal{O}(n)$ complexity, searching in a dictionary is on average only $\mathcal{O}(1)$ [16] → reduction of computing time
- The NumPy package is imported [9]. It efficiently implements support for arrays with arbitrary data types → reduction of memory space and computing time
- The adjacency matrix is very sparse and so only nonzero entries and their positions are stored in a sparse matrix (SciPy) [9] → reduction of memory space and computing time

Certainly there are more imaginable improvements, but their effects will be rather small due to the doubly exponential growing problem.

Results

Tables 2.1 and 2.2 show some first results of the computation for a MERA with $n = 2$ and $n = 4$ respectively. Depending on the overall number of time steps D, the second column of each table shows the length $l_{\max}$ of the longest string, the third the number $|V|$ of vertices in the graph and the fourth the overall number $|E|$ of edges. We analyze how these values depend on the size of a MERA, namely n and D. Let us start by calculating the number of unitaries and isometries at each time step: Left to the diagonal ($t < n$) we have $\#U_t = 2\#U_{t-2} + 3$ and to the right ($t \geq n$): $\#U_t = 2\#U_{t-2} - 3$. By recursion with the initial condition $\#U_1 = 3$ we obtain:

t even:

$$\begin{aligned} t < n: &\quad \#U_t = 3(2^{\frac{t+1}{2}} - 1) \\ t \geq n: &\quad \#U_t = 2^{\frac{t+1-n}{2}}(\#U_n - 3) + 3 = 3(2^{\frac{t+1}{2}}(1 - 2^{1-\frac{n}{2}}) + 1) \end{aligned} \tag{2.11}$$

D	$l_{\max}$	$\lvert V\rvert$	$\lvert E\rvert$
2	6	9	13
4	10	25	41
6	14	41	69
8	18	57	97

Table 2.1: MERA with $n = 2$ and the overall number of time steps D in the first column. The obtained length $l_{\max}$ of the longest string, the number of vertices $|V|$, and the number of edges $|E|$ are displayed in the following three columns.

D	$l_{\max}$	$\lvert V\rvert$	$\lvert E\rvert$
4	18	1833	6645
6	34	5478441	Memory Error
8	62	Memory Error	

Table 2.2: MERA with $n = 4$ and the overall number of time steps D in the first column. The obtained length $l_{\max}$ of the longest string, the number of vertices $|V|$, and the number of edges $|E|$ are displayed in the following three columns. The entries, which say "Memory Error" denote that the computation was not possible due to a lack of memory capacity.

The number of isometries $\#I_t$ is by one higher (left to the diagonal) or by one lower (right to the diagonal) than the number of unitaries:

t odd:

$$\begin{aligned} t < n: &\quad \#I_t = \#U_{t-1} + 1 = 3(2^{\frac{t}{2}} - 1) + 1 \\ t \geq n: &\quad \#I_t = \#U_{t-1} - 1 = 3(2^{\frac{t}{2}}(1 - 2^{1-\frac{n}{2}}) + 1) - 1 \end{aligned} \tag{2.12}$$

Knowing the number of unitaries at time step $t = D - 1$, the length $l_{\max}$ of the longest string is obtained by:

$$\begin{aligned} l_{\max} &= 2(D-n) + 2\#U_{D-1} = 2(D - n + 3(2^{\frac{D}{2}}(1 - 2^{1-\frac{n}{2}}) + 1)) \\ &= 2(D - n + 3) + 6(2^{\frac{D}{2}}(1 - 2^{1-\frac{n}{2}})) \end{aligned} \tag{2.13}$$

For $n = 2$, $l_{\max}$ depends only linearly on D ($l_{\max} = 2D + 2$). With the results from table 2.1 one sees that in this case the number of vertices $|V|$ and edges $|E|$ follow a linear dependence on the time D as well:

$$|V| = 8D - 7 \qquad\qquad |E| = 14D - 15 = \frac{7|V| - 11}{4} \tag{2.14}$$

Eigenvalues	0	0.78	2	2	2.46	4	4
Eigenvectors	$\begin{pmatrix}0.33\\0.33\\0.33\\0.33\\0.33\\0.33\\0.33\\0.33\\0.33\end{pmatrix}$	$\begin{pmatrix}0.88\\0.20\\-0.08\\-0.08\\-0.08\\-0.19\\-0.19\\-0.19\\-0.26\end{pmatrix}$	$\begin{pmatrix}0\\0\\0.57\\-0.36\\-0.21\\0.21\\0.36\\-0.57\\0\end{pmatrix}$	$\begin{pmatrix}0\\0\\0.09\\0.45\\-0.54\\0.54\\-0.45\\-0.09\\0\end{pmatrix}$	$\begin{pmatrix}-0.28\\0.40\\0.30\\0.30\\0.30\\-0.12\\-0.12\\-0.12\\0.67\end{pmatrix}$	$\begin{pmatrix}0\\0\\0.58\\-0.25\\-0.33\\-0.33\\-0.25\\0.58\\0\end{pmatrix}$	$\begin{pmatrix}0\\0\\-0.05\\0.52\\-0.47\\-0.47\\0.52\\-0.05\\0\end{pmatrix}$

Table 2.3: The 7 lowest eigenvalues and the corresponding eigenvectors of $L_{i,j}(G)$. As assumed the smallest eigenvalue is 0 and its eigenvector is a uniform superposition over all vertices.

The reason behind this behavior is the constant number of unitaries and isometries for all times. The $n = 2$ MERA is an exceptional case and does not show the unique growing property of a larger MERA. For $n = 4$ we are not able to write down the functional dependence of $|V|$ and $|E|$, because it was not possible to obtain enough values. However, we except a doubly exponential growth for the reasons that were explained before.
Although the program works only for very small MERA, for those we can confirm the 0 energy eigenvalue and the uniform superposition of valid strings as the ground state of the circuit Hamiltonian. As an example the results for a MERA of size $n = 2$ and $D = 2$ are shown and discussed in the following.
The vector V of valid strings corresponding to this MERA (figure 2.5) is:

$$V = \Big(0011 \quad 010101 \quad 100101 \quad 011001 \quad 010110 \quad 101001 \quad 100110 \quad 011010 \quad 101010\Big) \tag{2.15}$$

As the Laplacian matrix of the associated graph G we obtain:

$$L_{i,j}(G) = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 3 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 3 & 0 & -1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 3 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & -1 & 0 & 3 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & -1 & 0 & 3 & 0 & -1 \\ 0 & 0 & 0 & -1 & -1 & 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 3 \end{pmatrix} \tag{2.16}$$

The eigenvalues and corresponding eigenvectors of $L_{i,j}(G)$ are shown in table 2.3 above. The

smallest eigenvalue is 0 and its eigenvector is a uniform superposition over all vertices. This result confirms our calculations in section 2.2. Eigenvectors corresponding to higher eigenvalues are a linear combination with different weights on the vertices. There is still a noticeable structure: The eigenvalues 2 and 4 are degenerate and their eigenvectors have an "antisymmetric" structure (look at columns 3,4,6, and 7). For the other two displayed eigenvectors, some vertices still have the same weight in the linear combination, e.g. the value 0.30 occurs three times in the eigenvector of column 5. However, we could have already guessed such a behavior when examining the symmetries in the Laplacian matrix, or in the graph of figure 2.5.

2.3 Conclusion and Outlook

We were able to show that the space-time circuit-to-Hamiltonian construction can be applied to the MERA. However, it was not possible to numerically analyze the Laplacian matrix for a sufficiently large amount of different circuits. A computation up to a MERA size of about $n, D \sim 20$ would be ideal, our analysis reached out to a MERA size of only $n, D = 4$. Thus, we would require a memory space which is larger by roughly a factor of 2^{32} and therefore is not realizable. To avoid this problem, we could consider just a single layer of isometries and unitaries at the outermost fine-grained part of the MERA. Most of the valid strings lie at these time steps ($t \sim D$) and hence this region might serve as a good approximation for the whole circuit. Within this sub circuit of the MERA the analysis of this chapter can be repeated. The obtained Laplacian matrices and their spectra can then be studied in terms of symmetries, the eigenvalue gap, and its dependence on the size of the underlying MERA.
As mentioned in section 1.3.2, in [8] it is shown that the string in the formerly discussed circuit moves with constant velocity when it is time evolved with the propagating Hamiltonian (given by the adjacency matrix). In the MERA an exponentially increasing number of qubits is permanently mixed into the existing qubit string, so we cannot use the same argumentations as in [8]. Hence, it is also an interesting question how the forward propagation of a string in the MERA circuit behaves.

3 Entanglement Entropy

Entanglement is one of the most peculiar properties of quantum mechanics, especially because there does not exist a classical analog. In all quantum mechanical systems that involve more than one particle, entanglement can naturally occur and thus an understanding of it becomes crucial. The entanglement entropy is as a measure for the entanglement between two systems and therefore serves as an important quantity in the study of it.
Entanglement gives rise to new possibilities in computation theory, as for example the concept of quantum teleportation [13]. Besides the applications in quantum computing, the entanglement entropy can be used as a characterization of quantum mechanical systems. For example the scaling of entanglement entropy in critical systems differs from the one in non critical systems in a common way [6]. A tensor network should reflect the same properties as the system that it approximates. Hence, the scaling of entanglement entropy in tensor networks is an important question.
Beyond, entanglement seems to play a major role in a theory of quantum gravity. In [14] it is shown that gravity actually originates from entanglement renormalization. The MERA is a prominent example in which space-time (anti-de Sitter space) is induced by its holographic geometry and thus realizes an AdS/CFT correspondence. In [17] Verlinde interprets gravity as an entropic force that arises from changes in information when (massive) objects move relative to each other.
In the first section of this chapter we review the basic definitions of entropy and give some important properties that are needed for later calculations. The second section analyzes the entanglement entropy in the MERA (section 2.1) and in the circuit of section 1.2, which is called unitary circuit in the following. The scaling of entanglement entropy in both circuits obeys an area law, which is discussed as well. In the last section we turn to the entanglement entropy of a two dimensional space-time region in the unitary circuit and develop the steps to calculate it for arbitrary regions. An example is shown at the end. We conclude this chapter with an outlook on further research concerning this topic.

3.1 Definition and Properties

This section establishes the notation and important results needed for the subsequent sections and closely follows chapter 11 in reference [13]. Thus, for a more detailed introduction to the entropy I refer to [13]. The advanced reader can skip this section and can come back to it, if necessary.
In information theory the entropy is a measure for how uncertain the state of a system is, or in other words: it is the amount of information that is gained about the system when measuring its state. Both, a classical and quantum entropy can be defined. Let us start with the classical Shannon entropy.

Definition 3.1. *Shannon entropy*
The Shannon entropy of a classical system characterized by the random variable X whose n possible values occur with probabilities $p_1, ..., p_n$ is defined by:

$$H(X) \equiv H(p_1, ..., p_2) \equiv -\sum_{x=1}^{n} p_x \log_2 p_x \tag{3.1}$$

To define the quantum analog of the Shannon entropy, we first have to recapitulate the notions of the density matrix.

Definition 3.2. *Density matrix*
The density matrix of a quantum system that is in one of its states $|\Psi_i\rangle$ with probabilities p_i is given by:

$$\rho \equiv \sum_i p_i \, |\Psi_i\rangle \langle\Psi_i| \tag{3.2}$$

Definition 3.3. *Pure State*
The system is in a pure state, if its state $|\Psi\rangle$ is known and hence the density matrix can be written as $\rho = |\Psi\rangle \langle\Psi|$. If the state is unknown, the density matrix is called a mixed state.

Definition 3.4. *Reduced density matrix*
Suppose a composite system AB is in the state ρ^{AB}. Then the reduced density matrix of system A is obtained by:

$$\rho^A \equiv Tr_B(\rho^{AB}) \tag{3.3}$$

with Tr_B being the partial trace over system B.

Now we can define the quantum von Neumann entropy.

Definition 3.5. *Von Neumann entropy*
The von Neumann entropy is defined by:

$$S(\rho) \equiv -Tr(\rho \log_2 \rho) = -\sum_{x=1}^{n} \lambda_x \log_2 \lambda_x \tag{3.4}$$

where ρ is the density matrix of the system and λ_x are its eigenvalues.

From the following properties of the von Neumann entropy only selected ones will be proved. For all details I refer to [13].

Proposition 3.6. *Non-Negativity*
$S(\rho) \geq 0$ *with equality if and only if the system is in a pure state.*

Proposition 3.7. *Upper Bound $S(\rho) \leq \log_2 d$ with equality if and only if the system is in a completely mixed state $\mathbb{I}/d$.*

Proof. Let λ_i be the eigenvalues of ρ. Remember that $\sum_i \lambda_i = 1$. In the following we use that $-\log x \geq (1-x)/\ln 2$ with equality if and only if $x = 1$:

$$\begin{aligned} \log_2 d - S(\rho) &= \sum_i \lambda_i \log_2 d + \sum_i \lambda_i \log_2 \lambda_i \\ &= -\sum_i \lambda_i \log_2 \frac{1}{d\lambda_i} \\ &\geq \frac{1}{\ln 2} \sum_i \lambda_i \left(1 - \frac{1}{d\lambda_i}\right) \\ &= \frac{1}{\ln 2}(1-1) = 0 \end{aligned} \tag{3.5}$$

If and only if $\lambda_i = 1/d \; \forall i$, equality is correct. But these eigenvalues correspond to the completely mixed state $\rho = \mathbb{I}/d$. □

Proposition 3.8. *Entropy of a tensor product*
The entropy $S(\rho)$ of a tensor product $\rho = \rho_1 \otimes ... \otimes \rho_n$ is additive:

$$S(\rho) = S(\rho_1) + ... + S(\rho_n) \tag{3.6}$$

Proof. Let $\rho_i = U_i^\dagger \Lambda_i U_i$ where $\Lambda_i = \mathrm{diag}(\lambda_{i,1}, \lambda_{i,2}, ..., \lambda_{i,k_i})$ is the diagonal eigenvalue matrix of ρ_i. It follows: $\rho_1 \otimes ... \otimes \rho_n = (U_1 \otimes ... \otimes U_n)^\dagger \, \Lambda_1 \otimes ... \otimes \Lambda_n \, (U_1 \otimes ... \otimes U_n)$. Then the entropy $S(\rho)$ is obtained by: $S(\rho) = -\sum_{i=1}^{n} \sum_{j=1}^{k_i} \lambda_{i,j} \log_2 \lambda_{i,j} = \sum_{i=1}^{n} S(\rho_i)$ □

Proposition 3.9. *Suppose the system is in one of its states ρ_i with probabilities p_i and the states ρ_i have support on orthogonal subspaces. Then:*

$$S\left(\sum_i p_i\rho_i\right) = H(p_i) + \sum_i p_i S(\rho_i) \tag{3.7}$$

where $H(p_i)$ is the Shannon entropy defined before.

Proof. Let $\lambda_{i,j}$ be the eigenvalues of ρ_i. Then $\sum_i p_i\rho_i$ has eigenvalues $p_i\lambda_{i,j}$, since the ρ_i have support on orthogonal subspaces. Remember that $\sum_j \lambda_{i,j} = 1$. It follows:

$$\begin{aligned} S\left(\sum_i p_i\rho_i\right) &= -\sum_{i,j} p_i\lambda_{i,j}\log_2 p_i\lambda_{i,j} \\ &= -\sum_i p_i \log_2 p_i \sum_j \lambda_{i,j} - \sum_i p_i \sum_j \lambda_{i,j}\log_2\lambda_{i,j} \\ &= H(p_i) + \sum_i p_i S(\rho_i) \end{aligned} \tag{3.8}$$

□

Proposition 3.10. *Subadditivity*
Consider a composite system AB in the state ρ^{AB} with entropy $S(A,B) \equiv S(\rho^{AB})$. An upper and lower bound on this entropy is given by:

$$S(A,B) \leq S(A) + S(B) \tag{3.9}$$

$$S(A,B) \geq |S(A) - S(B)| \tag{3.10}$$

where $S(A) \equiv S(\rho^A) = S(Tr_B(\rho^{AB}))$ and $S(B) \equiv S(\rho^B) = S(Tr_A(\rho^{AB}))$.

Definition 3.11. *Entanglement Entropy*
Suppose a composite system AB is in a pure state ρ^{AB}. Then the entanglement entropy of system A is: $S(A) \equiv S(\rho^A) = Tr\left(\rho^A \log_2 \rho^A\right)$, where ρ^A is the reduced density matrix of subsystem A.

This definition implies that if the two subsystems are not entangled, and hence the density matrices ρ^A and ρ^B are pure states themselves, each entanglement entropy is zero.

Proposition 3.12.
If a composite system AB is in a pure state, the entropies of the two subsystems are equal: $S(A) = S(B)$.

Proof. By propositions 3.6 and 3.10: $0 = S(A,B) \geq |S(A) - S(B)| \Rightarrow S(A) - S(B) = 0$ □

Proposition 3.13.
Suppose a composite, pure system AB of qubits. Then the entanglement entropy $S(A)$ is upper bounded by the logarithm of the dimension $\dim(B)$ *of system B, which in turn equals the number n_B of qubits in system B:*

$$S(A) \leq \log_2(\dim(B)) = n_B \tag{3.11}$$

Proof. By proposition 3.12: $S(A) = S(B)$. Also by proposition 3.7 it follows that $S(B) \leq \log(\dim(B))$, which proves the first inequality. If system B consists of n_B qubits, its dimension is given by $\dim(B) = 2^{n_B}$. Hence: $\log_2(\dim(B)) = n_B$. □

3.2 Entanglement Entropy of a Region in Space

3.2.1 Unitary Circuit

For the unitary circuit discussed in section 1.2, we analyze how the entanglement entropy of a region in the circuit scales with the size of that region. First, we consider a connected region A_D in space, i.e. a line of qubits at time step t = D (see figure 3.1). The entanglement entropy $S(A_D)$ is obtained by the reduced density matrix $\rho_D^{A_D}$ of the qubits in the set A_D at time step $t = D$. In turn, this density matrix can be calculated by a unitary transformation from the density matrix $\rho_{D-1}^{A_{D-1}}$ one time step before:

$$\rho_D^{A_D} = \mathrm{Tr}_{\bar{A}_D}\left(U\rho_{D-1}^{A_{D-1}}U^\dagger\right) \tag{3.12}$$

The region A_{D-1}, displayed by the green dots at $t = D-1$ in figure 3.1, is the set of only those qubits that can influence qubits in A_D. In general the region A_{D-1} contains more qubits than A_D. These excess qubits form a new set $\bar{A}_D$ and are illustrated by two red dots at time $t = D$ in figure 3.1. It follows that $A\bar{A}_D \equiv A_D \cup \bar{A}_D = A_{D-1}$. The qubits in $\bar{A}_D$ are not part of the considered region and therefore traced out in equation 3.12. We can repeat this procedure with the density matrix $\rho_{D-1}^{A_{D-1}}$ and the following ones until we reach the density matrix $\rho_0^{A_0}$ at the initial time step $t = 0$:

$$\rho_0^{A_0} \overset{U\rho U^\dagger}{\Longrightarrow} \rho_1^{A\bar{A}_1} \overset{\mathrm{Tr}_{\bar{A}}}{\Longrightarrow} \rho_1^{A_1} \overset{U\rho U^\dagger}{\Longrightarrow} ... \overset{\mathrm{Tr}_{\bar{A}}}{\Longrightarrow} \rho_{T-1}^{A_{T-1}} \overset{U\rho U^\dagger}{\Longrightarrow} \rho_T^{A\bar{A}_T} \overset{\mathrm{Tr}_{\bar{A}}}{\Longrightarrow} \rho_T^{A_T} \tag{3.13}$$

Starting from the initial state $\rho_0^{A_0}$, we are able to calculate the density matrix $\rho_D^{A_D}$, if all unitary gates in the circuit are known, and hence we can obtain the entropy $S(A_D)$. But rather of computing the entropy explicitly, we want to find an upper bound for $S(A_D)$ by exploiting its properties defined in the section before. Consider the difference of the entropy $S(A_D)$ of region A_D at time $t = D$ and the entropy $S(A_0)$ of the initial region A_0 at time $t = 0$:

$$\begin{aligned} S(A_D) - \underbrace{S(A_0)}_{=0} &= \overbrace{S(A_D) - S(A_{D-1})}^{\Delta S_D} + \overbrace{S(A_{D-1}) - S(A_{D-2})}^{\Delta S_{D-1}} + S(A_{D-2}) - ... \\ &= \sum_{t=1}^{D} \Delta S_t \end{aligned} \tag{3.14}$$

At $t = 0$ the system is in a tensor product state with no entanglement. Thus, the entropy is zero by proposition 3.6.
Furthermore, a unitary transformation of a density matrix leaves its entropy invariant:

$$S(A\bar{A}_t) = S(U\rho_{t-1}^{A_{t-1}}U^\dagger) = S(A_{t-1}) \tag{3.15}$$

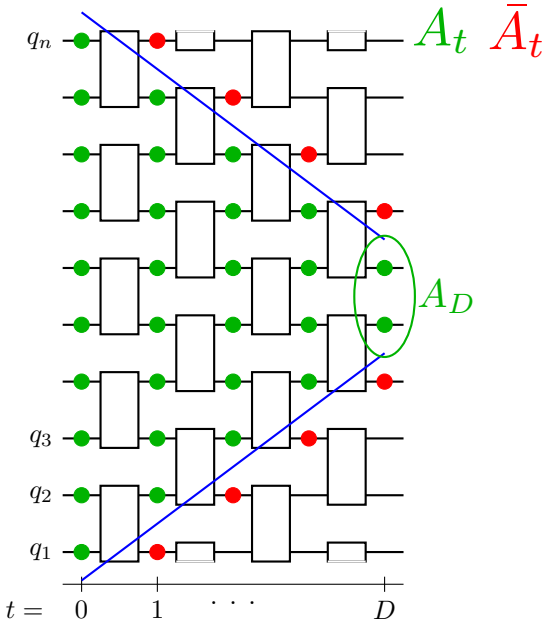

Figure 3.1: In this example the region A_D is given by the two qubits (green dots) at time step $t = D$. The set A_{D-1} contains the four qubits (green dots) one time step earlier at $t = D - 1$. The red dots denote the qubits that are traced out in order to get from the density matrix of region A_{D-1} to the reduced density matrix of region A_D. These two red dots at $t = D$ build the set $\bar{A}_D$. For earlier time steps the regions A_t and $\bar{A}_t$ are again defined by the "green" or "red" qubits at these times respectively. The entropy $S(A_D)$ of region A_D is upper bounded by the length of its causal cone (blue lines).

Remember that $A\bar{A}_t \equiv A_t \cup \bar{A}_t = A_{t-1}$ are sets of qubits. (The arguments of the entropies are just a shorthand for the corresponding density matrices.)
From propositions 3.7 and 3.10 we obtain:

$$S(A\bar{A}_t) \geq S(A_t) - \underbrace{S(\bar{A}_t)}_{\leq \log(\dim \bar{A}_t)} \tag{3.16}$$

Inserting equation 3.15 into 3.16 yields a relation for the entropy of region A_{t-1} and A_t:

$$\begin{aligned} &S(A_{t-1}) \geq S(A_t) - \log(\dim \bar{A}_t) \\ \Rightarrow &\Delta S_t = S(A_t) - S(A_{t-1}) \leq \log(\dim \bar{A}_t) \end{aligned} \tag{3.17}$$

Next, we can apply proposition 3.13 to the last equation, obtaining $\Delta S_t = S(A_t) - S(A_{t-1}) \leq \log(\dim \bar{A}_t) = n_t^{\mathrm{Tr}}$, where n_t^{Tr} are the number of qubits that are traced out at time step t. For the unitary circuit in one dimension, these are always the 2 qubits (red dots) at the boundary of the causal cone. In total, the entropy of region A_D (equation 3.14) can be expressed by:

$$S(A_D) = \sum_{t=1}^{D} \Delta S_t \leq \sum_{t=1}^{D} n_t^{\mathrm{Tr}} \leq 2D = \text{const.} \tag{3.18}$$

The entropy $S(A_D)$ is upper bounded by twice the time depth D of the circuit. Thus, it scales as the length of its past causal cone. For constant depth circuits the entropy is constant and does not depend on the size of the region A_D.
In the following, we study the case in which region A_D consists of two disconnected regions B

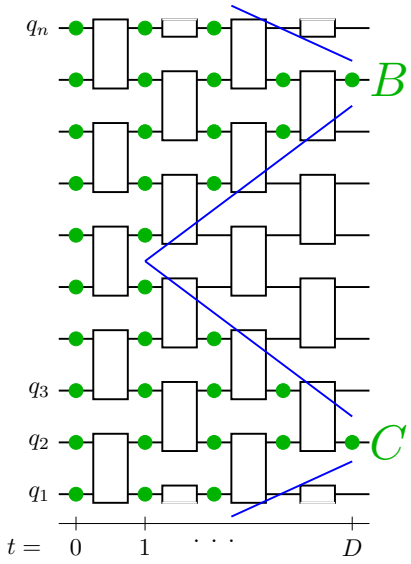

Figure 3.2: An example of region A_D consisting of two disconnected regions B and C that contain each only one qubit. The entropy $S(B, C)$ is upper bounded by the length of the resulting causal cone, displayed by the blue lines.

and C ($A_D = B \cup C$), as for example in figure 3.2. Proposition 3.10 gives both an upper and lower bound for the entropy $S(A_D) = S(B, C)$:

$$S(B, C) \leq S(B) + S(C) \tag{3.19}$$

$$S(B, C) \geq |S(B) - S(C)| \tag{3.20}$$

First, consider the simple case that regions B and C have non intersecting causal cones, which implies that their qubits could not influence each other. Then the system BC can be written as a tensor product state $B \otimes C$ and according to proposition 3.8: $S(B, C) = S(B) + S(C)$. The upper bound from equation 3.18 is twice as high, because at each time step 4 qubits have to be traced out (2 for each causal cone): $n_t^{\mathrm{Tr}} = 4$. However, if the causal cones of the two subsystems intersect (figure 3.2), the equality is not true in general. At all time steps, where the backwards causal cones of the two subsystems merged, the number n_t^{Tr} of qubits to trace out is still 2, while for the other regime $n_t^{\mathrm{Tr}} = 4$. Suppose the causal cones merge at time step $t = t'$. Then the upper bound for the entropy $S(B, C)$ is calculated by:

$$S(B, C) = \sum_{t=1}^{D} n_t^{\mathrm{Tr}} \leq 2t^{'} + 4(D - t^{'}) \tag{3.21}$$

Again, the result shows that the entropy $S(B, C)$ scales with the length of the merged causal cones of regions B and C.

Let us examine how the result from equation 1.2 is changed in the case of a two dimensional circuit. The only difference arises due to another number n_t^{Tr} of qubits one has to trace out per time step, since the rest of the derivation above did not depend on the circuit dimension. For simplicity suppose that region A_D is a circle area with radius r_D. The backwards causal cone is now three dimensional with linearly in t decreasing radius r_t: $r_t = r_D + (D - t)$. For each

time step the qubits which have to be traced out lie again at the boundary of the causal cone. Therefore, their number is proportional to the momentary perimeter: $n_t^{\mathrm{Tr}} \sim 2\pi r_{t-1}$. Inserting this into equation 3.14 yields to:

$$\begin{aligned} S(A_D) \leq \sum_{t=1}^{D} n_t^{\mathrm{Tr}} \sim \sum_{t=1}^{D} 2\pi r_{t-1} &= \sum_{t=1}^{D} 2\pi (r_D + (D - t + 1)) \\ &= 2\pi D(r_D + D + 1) - 2\pi \sum_{t=1}^{D} t \\ &= 2\pi D \left(r_D + \frac{D+1}{2} \right) \\ &= \mathcal{O}(r_D) \end{aligned} \tag{3.22}$$

For two dimensional constant depth circuits, the entropy scales with the radius, or equivalently with the perimeter of the considered region. Thus, the entanglement entropy is not constant, opposed to the one dimensional case.

3.2.2 MERA

In this section we consider the scaling of entanglement entropy in the MERA, which was presented in section 2.1. The analysis follows reference [6] in a close manner.
The major difference between the MERA and the unitary circuit is the behavior of the past causal cone. The causal cones of regions in the MERA have a bounded width, which means that their size is independent of the length of the considered region. This attribute is explained by the isometries and their shrinking property.
Let us examine the one dimensional MERA in figure 3.3 c). We again define a region B, with length l_0, as a set of qubits at the last time step $t = D$. A scale parameter z is introduced that corresponds to the different length scales in the MERA: $z = 0$ represents the smallest length scale and thus corresponds to time step $t = D$. For every two time steps going backwards, the scale parameter increases by one, e.g. $z = 1$ corresponds to $t = D - 2$. The past causal cone $C(B)$ of region B is displayed by the purple shaded area. Going backwards in time unitaries will always enlarge the width l of the causal cone by at most 2 qubits, while isometries always reduce its width by approximately a factor of 2. We have to distinguish between two cases: If the width of the causal cone at scale z is small, namely $l_z \leq 4$, the unitaries will always add about the same number of qubits as the isometries remove, leaving the width l nearly constant (it alternates between 3 and 4 qubits): $l_{z+1} \sim l_z$ (figure 3.3 b). This regime of a causal cone is called stationary. On the other hand, if $l_z > 4$, the isometries dominate and the width l is divided roughly in halves: $l_{z+1} \sim l_z/2$ (figure 3.3 a). At scale $z = 0$ the width of $C(B)$ equals the length l_0 of region B. Hence, the transition from shrinking regime to stationary regime occurs at the scale $\bar{z} \sim \log_2(l_0)$.
In analogy to the previous section concerning the unitary circuit, we consider differences ΔS_z

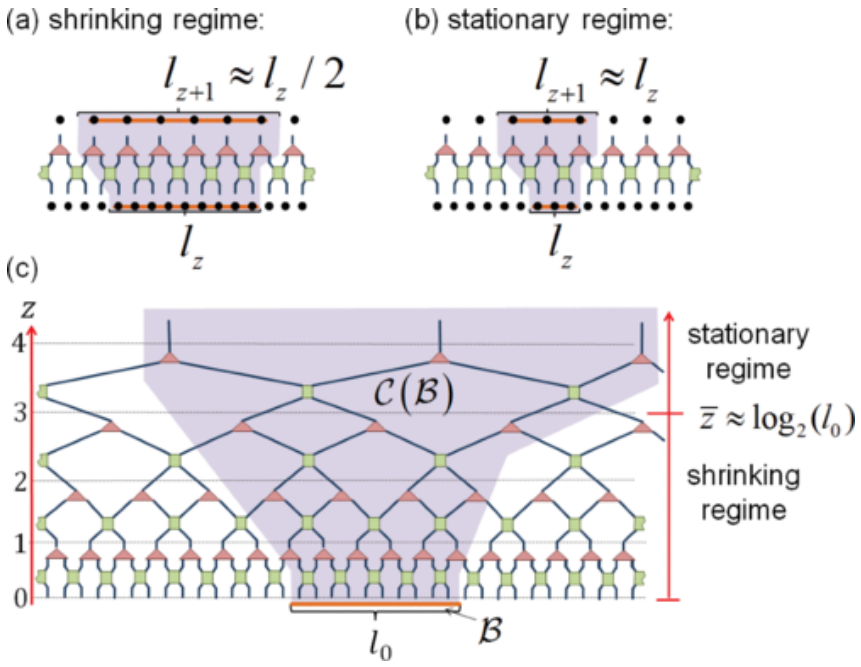

Figure 3.3: (Taken from [6]). a) In the shrinking regime of the causal cone $C(B)$ of a region B, its width l decreases exponentially with the scale parameter z. b) In the stationary regime the width l of the causal cone stays about constant. c) The causal cone $C(B)$ of region B is illustrated by the purple shaded region. The transition from shrinking to stationary regime occurs at a scale $\bar{z} \sim \log_2(l_0)$.

of the entropies at two successive scales. However, this time the sum over the differences ΔS_z should end at the transition scale $\bar{z}$. At each time step at most 2 qubits have to be traced out. Hence, at each scale, which involves two time steps, there are at most 4 qubits to trace out leading to $n_z^{\mathrm{Tr}} = 4$:

$$\begin{aligned} S(B) - \underbrace{S(B_{\bar{z}})}_{\leq \log 2^4 = 4} &= \overbrace{S(B) - S(B_{z+1})}^{\Delta S_{z=0}} + \overbrace{S(B_{z+1}) - S(B_{z+2})}^{\Delta S_{z=1}} + S(B_{z+2}) - \dots \\ &= \sum_{z=0}^{\bar{z}-1} \Delta S_z \leq \sum_{z=0}^{\bar{z}-1} n_z^{\mathrm{Tr}} \leq 4\bar{z} = \mathcal{O}(\log_2(l_0)) \end{aligned} \tag{3.23}$$

We know that the number of qubits inside the causal cone at scale $\bar{z}$ is at most 4 and thus can upper bound the entropy $S(B_{\bar{z}})$ above by a constant (proposition 3.7). Therefore, in a one dimensional MERA the entropy of a region B with length l_0 scales as the logarithm of l_0:

$$S(B) = \mathcal{O}(\log_2(l_0)) \tag{3.24}$$

In the preceding calculations we have not considered one possibility, namely that $\bar{z} \sim \log(l_0)$ lies outside the circuit, and hence there is no stationary regime of the causal cone. In this case

the entropy is independent of the length l_0, as in the unitary circuit. However, in the following I assume that the MERA involves infinite time steps (infinite number of scales), so that equation 3.24 always holds.
So far we considered the one dimensional MERA. Let us briefly study the two dimensional case. In the shrinking regime the radius r of the three dimensional past causal cone decreases exponentially with scale z: $r_z \sim r_0 2^{-z}$. The qubits, which are traced out at each scale lie at the boundary of the causal cone and thus decrease exponentially as well: $n_z^{\mathrm{Tr}} \sim 2\pi r_0 2^{-z}$. Therefore, equation 3.23 is changed in the following way:

$$\begin{aligned} S(B) \leq \sum_{z=0}^{\bar{z}-1} n_z^{\mathrm{Tr}} \sim 2\pi r_0 \sum_{z=0}^{\bar{z}-1} 2^{-z} &= 2\pi r_0 \frac{1-2^{-\bar{z}}}{1-2^{-1}} \\ &= 4\pi r_0 (1 - 2^{-\log_2(2r_0)}) \\ &= 4\pi r_0 - 2\pi \\ &= \mathcal{O}(r_0) \end{aligned} \tag{3.25}$$

In the two dimensional MERA the entanglement entropy $S(B)$ scales as the perimeter of region B. Thus, it displays the same dependence as the unitary circuit in two dimensions.

3.2.3 Area Law

This section is supposed to summarize the results from the previous sections and interpret them in terms of an area law for entanglement entropy.
If we consider the entanglement entropy $S(A)$ of a region A in the ground state of a local system, one observes that the entropies $S(A)$ scale in a common way, which differs from the behavior of excited states. For ground states the entropy $S(A)$ scales as the size $|\partial A|$ of the boundary of region A, rather than its volume [6]:

$$S(A) = \mathcal{O}(|\partial A|) \tag{3.26}$$

This equation is known as the area law for entanglement entropy. The entropy in the constant depth unitary circuit follows this area law and thus this circuit is a candidate to correctly describe ground states of such systems in at least one and two dimensions.
One exception of the area law is given by quantum critical systems, which display a logarithmic correction [6]:

$$S(A) = \mathcal{O}(l^{d-1} \log_2(l)) \tag{3.27}$$

d denotes the dimension of the system and l some common length in the region A, for example its radius. The binary MERA in one dimension obeys this adjusted area law. In reference [6] a slightly modified MERA, called branching MERA, is considered that adapts the scaling law

Dimension	$S \sim l^{d-1}$	$S \sim l^{d-1} \log_2(l)$
$d = 1$	Unitary	MERA
$d = 2$	Unitary, MERA	branch. MERA
$d = 3$	(Unitary, MERA)	branch. MERA

Table 3.1: The ground states of non-critical systems obey a simple area law for their entanglement entropy. The unitary circuit shows the same behavior as those systems. Although the case $d = 3$ was not discussed, we expect the same dependence for this dimension as well. The MERA can be used as a description of quantum critical systems. In one dimension its entropy scales according to an adjusted area law, which involves a logarithmic correction. The branching MERA displays the logarithmic correction also for higher dimensions [6].

of equation 3.27 for higher dimensions as well. By analyzing further properties of the MERA it turns out that the MERA in general is a good approximation for a number of quantum critical systems [5].
Table 3.1 summarizes the results. Although we did not show it explicitly, one expects a behavior in three dimensions as shown in the last row of the table. In the case of the MERA this can be verified in [6].

3.3 Entanglement Entropy of a Space-Time Region

In this and the subsequent sections we analyze the entanglement entropy of a two dimensional space-time region in the unitary circuit. Some parts of this analysis are extracted from [15].
To regard the circuit as a two dimensional system, it is convenient to use the grid representation of section 1.3.2. In this case each edge e is characterized by a spatial position $x \in \{1, ..., 2n\}$ that corresponds to a qubit q_x, and a position in time $t \in \{0, ..., D\}$. The states associated to each edge can be encoded in a dual rail: If an edge e is not occupied by a qubit, its state is given by $|00\rangle_e$. If the edge is occupied by a qubit with internal state $|0\rangle$, the corresponding state is $|10\rangle_e$ and if the internal qubit state reads $|1\rangle$, the state associated to that edge is $|01\rangle_e$. Linear combinations of the last two possibilities can occur as well. In this way we can represent the internal qubit state and its time together in just one state. Be aware that the state $|11\rangle$ is not valid in this description.
A space-time region A is defined by a set of edges $e = (x, t)$. Figure 3.4 shows an example of such a region, colored blue. The boundary (red) of the region should clearly distinguish between edges inside A and outside, denoted by $\bar{A}$. Therefore, the boundary line of region A is the set of all vertices, which connect edges $e \in A$ with edges $e \in \bar{A}$.
The entanglement entropy of a region A is calculated by its reduced density matrix which is in turn obtained by tracing over all edges that are not included in A. Opposed to the section before, this trace taking now also involves a tracing over times t. The required tools for this procedure are established in the following.

3.3.1 Tracing over time

In this section we work out what is meant by tracing over time. First, we regard only the time register. At the end of this section we determine what has to be changed when involving the qubit states.
As defined in section 1.2 a string or valid time configuration $|s\rangle$ is the tensor product of time states of all qubits $q_1, ..., q_{2n}$:

$$|s\rangle = |t_1^s\rangle \otimes |t_2^s\rangle \otimes ... \otimes |t_{2n}^s\rangle \tag{3.28}$$

Each time state $|t\rangle$ can be represented by a pulse clock (section 1.3.1):

$$\begin{aligned} t \in \{0, ..., D\} \\ |t = 0\rangle &= |\underbrace{1000...0}_{\text{length } D+1}\rangle \\ |t = i\rangle &= |0...0\underbrace{1}_{i+1}0...0\rangle \\ |t = D\rangle &= |0...001\rangle \end{aligned} \tag{3.29}$$

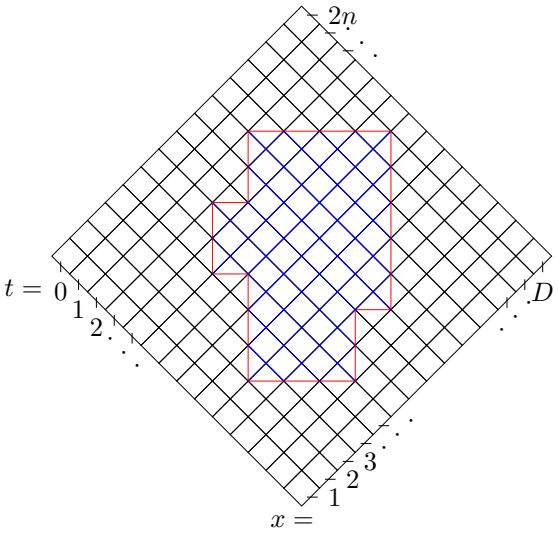

Figure 3.4: Example of a space-time region A in the unitary circuit. Region A contains all edges $e = (x, t)$ that are colored blue. Each spatial position x corresponds to one qubit q_x of the circuit. The boundary line (red) separates edges in A from edges in $\bar{A}$.

A state $|t\rangle$ is obtained by $D+1$ qubits from which exactly one is not in the state $|0\rangle$. I call these time qubits, in order to not confuse them with the qubits on which the actual gates are applied.

Definition 3.14. *Tr_t (tracing over time t) means tracing out time qubit $t+1$.*

We can distinguish three different cases that directly follow from the definition ("_" denotes the time qubit that is traced out):

Proposition 3.15. *The rules for tracing over a time t*

$$\begin{aligned} &1) && Tr_t |t\rangle\langle t| = |0...0\underbrace{\_}_{t+1}0...0\rangle\langle 0...0\_0...0| \\ &2)\ t' \neq t: && Tr_t |t\rangle\langle t'| = 0 \\ &3)\ t_1 \neq t, t_2 \neq t: && Tr_t |t_1\rangle\langle t_2| = |t_1\rangle\langle t_2| \end{aligned} \tag{3.30}$$

The following example illustrates the results:

Example:
Consider $|t = 3\rangle = |000100\rangle$. Then:

$$\mathrm{Tr}_{t=3} |t = 3\rangle\langle t = 3| = |000\_00\rangle\langle 000\_00| \tag{3.31}$$

After taking the trace, all time qubits left are in the state $|0\rangle$.

$$\mathrm{Tr}_{t=3} |t = 1\rangle\langle t = 4| = |010\_00\rangle\langle 000\_10| = |t = 1\rangle\langle t = 4| \tag{3.32}$$

In this case the time qubit which points to the time is not traced out and we obtain the initial state.

Next, we determine how tracing over a set of times $T = \{t_{\text{in}}, ..., t_{\text{out}}\}$ is performed:

Proposition 3.16. *The rules for tracing over a set of times* $T = \{t_{in}, ..., t_{out}\}$

$$\begin{aligned}
&1)\ t \in T: && Tr_T |t\rangle \langle t| = |0...0 \underbrace{__}_{t_{in}+1} _____ \underbrace{__}_{t_{out}+1} 0...0\rangle \langle 0....0_____0...0| \\
&2)\ t \in T, t \neq t': && Tr_T |t\rangle \langle t'| = 0 \\
&3)\ t, t' \notin T: && Tr_T |t\rangle \langle t'| = |t\rangle \langle t'|
\end{aligned} \tag{3.33}$$

The rules follow directly from definition 3.14 and proposition 3.15. In the first case, it is not possible to restore the initial state. The information about the time is lost.
Let us apply the established tracing rules to a density matrix of the following form:

$$\tilde{\rho} = \frac{1}{N} \sum_{s,s'} |s\rangle \langle s'| \tag{3.34}$$

It is the history state of the circuit excluding the data register. s and s' are valid time configurations and N is a normalization constant. Given a region A, for each qubit q_x there is a region T_x in time that does not lie in A and has to be traced out. However, these set of times $T_x = \{t_{\text{in}}, ..., t_{\text{out}}\} \notin A$ can be different for every qubit q_x. Thus, an expression for $\left(\prod_{x=1}^{2n} \text{Tr}_{T_x \in \bar{A}}\right) |s\rangle \langle s'|$ needs to be figured out.

Case 1: (Figure 3.5) s completely lies outside of region A, i.e. $t_x^s \in \bar{A}\ \forall x$. Then by rules 1 and 2, s' has to be equal to s ($t_x^s = t_x^{s'}\ \forall x$):

$$\left(\prod_{x=1}^{2n} \text{Tr}_{T_x \in \bar{A}}\right) |s\rangle \langle s'| = \delta_{s,s'} \, |\_\_\_\_0...0\_\_\_\_\rangle \langle \_\_\_\_0...0\_\_\_\_|^{2n} \tag{3.35}$$

Case 2: s lies completely inside the region A, i.e. $t_x^s \in A\ \forall x$. We exclude this case, since the region A should not range over the whole circuit. Hence, there are always parts of each string that are not in region A.
Case 3: (Figure 3.6) s runs partly inside the regions A and $\bar{A}$. Then by rules 2 and 3: $\forall x$ for which $t_x^s \in A : t_x^{s'} \in A$ and $\forall x$ for which $t_x^s \notin A : t_x^{s'} = t_x^s$. The string segment with $t_x^s \in A$ is denoted by s_A, the string segment with $t_x^s \notin A$ by $s_{\bar{A}}$, so that $|s\rangle = |s_A s_{\bar{A}}\rangle$.

$$\left(\prod_{x=1}^{2n} \text{Tr}_{T_x \in \bar{A}}\right) |s = s_A s_{\bar{A}}\rangle \langle s' = s'_A s'_{\bar{A}}| = \delta_{s_{\bar{A}}, s'_{\bar{A}}} \underbrace{|\_\_\_\_0...0\_\_\_\_\rangle \langle \_\_\_\_0...0\_\_\_\_|}_{\text{for every } t \in s_{\bar{A}}} \otimes |s_A\rangle \langle s'_A| \tag{3.36}$$

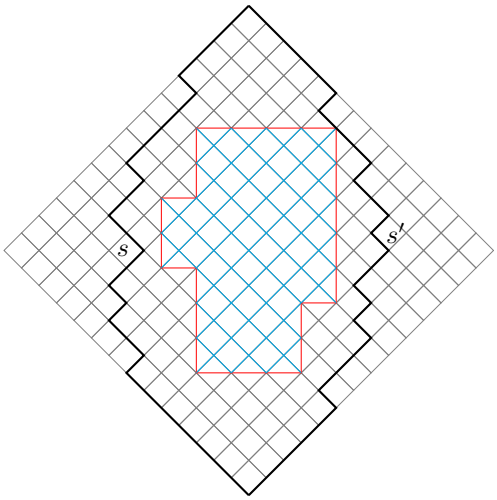

Figure 3.5: Example of strings s and s' that run completely outside region A. After tracing over all times in $\bar{A}$, there is no information about the time states left. The result is only nonzero, if strings s and s' in the state $|s\rangle\langle s'|$ are equal.

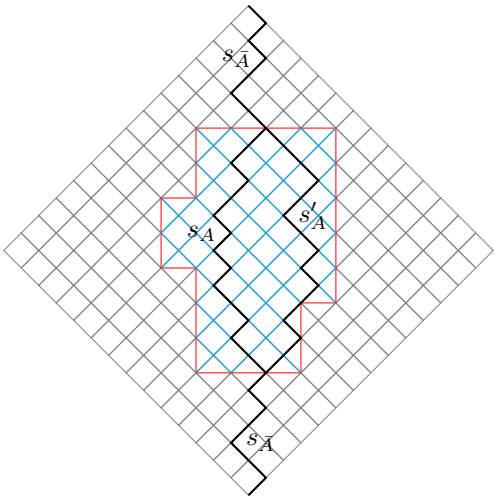

Figure 3.6: Example of strings s and s' that run partly inside region A. The string segments $s_{\bar{A}}$ outside A are traced out. Again the result is only nonzero if these string segments of s and s' in the state $|s\rangle\langle s'|$ are equal. However, inside region A the string segments s_A and s'_A can continue on different paths.

When applying these cases on the expression in equation 3.34, we obtain:

$$\begin{aligned}\left(\prod_{x=1}^{2n}\mathrm{Tr}_{T_x\in\bar{A}}\right)\tilde{\rho} &= \frac{1}{N}\left(\prod_{x=1}^{2n}\mathrm{Tr}_{T_x\in\bar{A}}\right)\sum_{s,s'}|s=s_As_{\bar{A}}\rangle\langle s'=s'_As'_{\bar{A}}| \\ &= \frac{1}{N}\Bigg(\sum_{s,s'\notin A}\delta_{s,s'}\,|\_\_\_\_0...0\_\_\_\_\rangle\langle\_\_\_\_0...0\_\_\_\_|^{2n} \\ &\qquad + \sum_{\substack{s,s'\\ \text{partly in } A}}\delta_{s_{\bar{A}},s'_{\bar{A}}}\underbrace{|\_\_\_\_0...0\_\_\_\_\rangle\langle\_\_\_\_0...0\_\_\_\_|}_{\text{for every } t\in s_{\bar{A}}}\otimes|s_A\rangle\langle s'_A|\Bigg)\end{aligned} \tag{3.37}$$

The strings s, which partly lie inside the region A, cut the boundary line at least twice. The vertices v, at which a string intersects the boundary line, are called boundary vertices. Each string can be characterized by the set V of its boundary vertices. For strings which run completely outside of A, this set is empty: $V=\emptyset$. For strings which cut the boundary line exactly twice, the set V involves two vertices: $V=\{v_1,v_2\}$, for strings that cut it four times, there are four boundary vertices: $V=\{v_1,...,v_4\}$, and so on. Figure 3.7 illustrates two examples of strings and their boundary vertices. The sums over s and s' in equation 3.37 can be split up in a sum

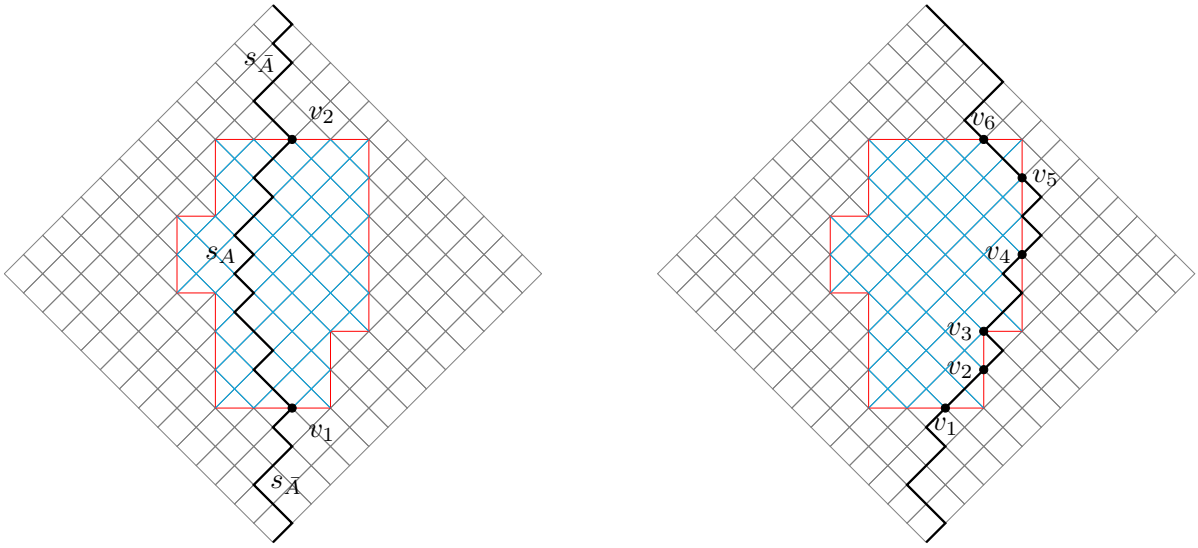

Figure 3.7: Example of a boundary vertex set V containing two (left) and six (right) vertices $V = \{v_1, v_2\}$ or $V = \{v_1, ..., v_6\}$. Whenever a string s goes from the region A to the region $\bar{A}$ or vice versa, the intersection point of the boundary line marks one boundary vertex v. In the figure on the right the string segments s_A and $s_{\bar{A}}$ are given by multiple, disconnected string segments.

over all possible boundary vertex sets V, and a sum over strings s^V and s'^V that intersect the boundary line in exactly these vertices:

$$\left(\prod_{x=1}^{2n} \mathrm{Tr}_{T_x \in \bar{A}}\right) \tilde{\rho} = \frac{1}{N}\Bigg(\sum_{s,s' \notin A} \delta_{s,s'} \, |0...0\_\_\_\_0...0\rangle \langle 0...0\_\_\_\_0...0|^{2n} + \sum_{V \neq \emptyset} \sum_{s^V, s'^V} \delta_{s^V_{\bar{A}}, s'^V_{\bar{A}}} \underbrace{|0...0\_\_\_\_0...0\rangle \langle 0...0\_\_\_\_0...0|}_{\text{for every } t \in s_{\bar{A}}} \otimes |s^V_A\rangle \langle s'^V_A| \Bigg) \tag{3.38}$$

The second sum should only involve nonempty sets $V \neq \emptyset$, since the case of an empty boundary vertex set is considered separately in the first term.
So far we only traced over a region in time. But the actual history state of the circuit involves a qubit register and all qubits that occupy edges outside of region A have to be traced out as well. Thus, we now trace over all edges $e = (x, t)$ outside of A. Each trace over a time t is followed by tracing out all qubits q_x being at that time. The rules from this section have to be modified slightly:

Proposition 3.17. *Tracing over all edges in* $E = \{(x, t_{in}), ..., (x, t_{out})\}$*:*

$$\begin{aligned}
&1)\ (x,t) \in E: && Tr_E\, |q_{x,t}\rangle \langle q_{x,t}| \otimes |t\rangle \langle t| = |vac\rangle \langle vac| \\
&2)\ (x,t) \in E, t \neq t': && Tr_E\, |q_{x,t}\rangle \langle q_{x,t'}| \otimes |t\rangle \langle t'| = 0 \\
&3)\ (x,t), (x,t') \notin E: && Tr_E\, |q_{x,t}\rangle \langle q_{x,t'}| \otimes |t\rangle \langle t'| = |q_{x,t}\rangle \langle q_{x,t'}| \otimes |t\rangle \langle t'|
\end{aligned} \tag{3.39}$$

We renamed the resulting state in equation 3.39 as $|\text{vac}\rangle$ (vacuum state), because there is no information left, it was completely traced out. The subscript t of the qubit state $|q_{x,t}\rangle$ implies that it depends in general on the time step t. After each time step a unitary operation is performed on the qubits, changing their internal states. Furthermore, be aware that in the dual rail encoding a qubit and its clock are represented together and hence the tensor product symbol should not be used. However, for readability I hold on to this notation.
With all these tools at hand we are now able to calculate the reduced density matrix of a space-time region A in the circuit and its resulting entropy.

3.3.2 Calculating the Entropy

The history state of the unitary circuit is:

$$|\Psi_{\text{history}}\rangle = \frac{1}{\sqrt{N}} \sum_s |\xi_s\rangle \otimes |s\rangle \tag{3.40}$$

where

$$|\xi_s\rangle = V(s_{\text{init}} \to s)\, |\xi\rangle \tag{3.41}$$

$|\xi\rangle$ denotes the initial state of all qubits: $|\xi\rangle = |q_{1,t=0}\rangle \otimes |q_{2,t=0}\rangle \otimes ... \otimes |q_{2n,t=0}\rangle$. The circuit system is supposed to be in its history state and hence the corresponding density matrix is:

$$\rho_{\text{history}} = |\Psi_{\text{history}}\rangle \langle \Psi_{\text{history}}| = \frac{1}{N} \sum_{s,s'} |\xi_s\rangle \langle \xi_{s'}| \otimes |s\rangle \langle s'| \tag{3.42}$$

We want to compute the reduced density matrix ρ_A of a region A. Therefore we trace over all edges $e = (x, t)$ outside of A:

$$\rho_A = \text{Tr}_{e \in \bar{A}}\ \rho_{\text{history}} \tag{3.43}$$

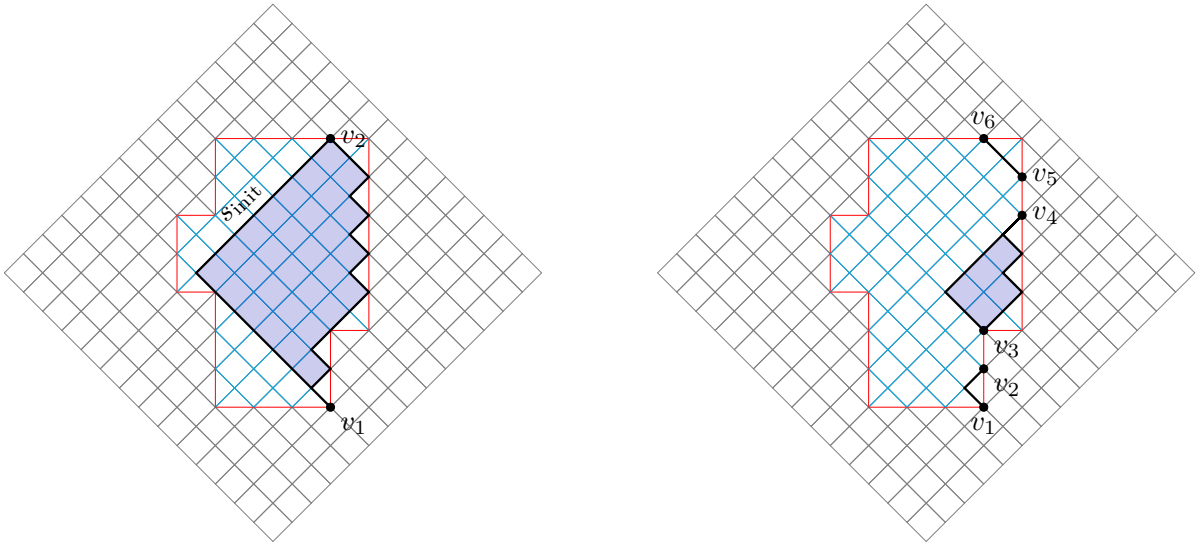

Figure 3.8: The sub circuits corresponding to the boundary vertex set $V = \{v_1, v_2\}$ (left) and $V = \{v_1, ..., v_6\}$ (right) are enclosed by black lines. The left black line represents the initial string of the sub circuit. From that, all other strings which are compatible with the boundary vertices in V, are obtained by unitary evolution within the sub circuit. In the figure on the right the sub circuit consists of three disconnected sub circuits. The sub circuits corresponding to vertices v_1, v_2 and v_5, v_6 are trivial and contain no dynamics at all.

With the results from the last section (especially equation 3.38 and proposition 3.17), we obtain:

$$\rho_A = \frac{1}{N}\left(\sum_{s\notin A} |\text{vac}\rangle\langle\text{vac}| + \sum_{V\neq\emptyset}\sum_{s^V,s'^V} \underbrace{\delta_{s^V_{\bar{A}},s'^V_{\bar{A}}}\left(\text{Tr}_{q|s_{\bar{A}}} |\xi_{s^V}\rangle\langle\xi_{s'^V}|\right)}_{=\rho^V_{s_A,s'_A}} \otimes |s^V_A\rangle\langle s'^V_A|\right) \tag{3.44}$$

$\text{Tr}_{q|s_{\bar{A}}}$ is the trace over all qubits whose times are in $s^V_{\bar{A}}$, thus in the region outside of A.
We can further simplify equation 3.44 by introducing the probabilities $\text{prob}(V = \emptyset)$ of finding a string running completely outside of region A, and $\text{prob}(V)$ of having a string with boundary vertex set V. It directly follows that $\sum_{V\neq\emptyset}\text{prob}(V) = 1 - \text{prob}(V = \emptyset)$. Let $N(s^V_A)$ and $N(s^V_{\bar{A}})$ be the number of possible string segments in A and $\bar{A}$ respectively so that $\text{prob}(V) = \frac{N(s^V_{\bar{A}})N(s^V_A)}{N}$.

$$\rho_A = \text{prob}(V = \emptyset)\,|\text{vac}\rangle\langle\text{vac}| + \sum_{V\neq\emptyset}\text{prob}(V)\frac{1}{N(s^V_A)}\sum_{s^V_A,s'^V_A}\rho^V_{s_A,s'_A}\otimes|s^V_A\rangle\langle s'^V_A| \tag{3.45}$$

Given a specific set of boundary vertices V, we are left with a sum over states which very much looks like a history state, but restricted to a smaller part of the circuit. All string segments s^V_A inside A, which are compatible with V, form a new sub circuit to which we can associate a Hamiltonian H^V. Two examples of such sub circuits are shown in figure 3.8. In complete analogy

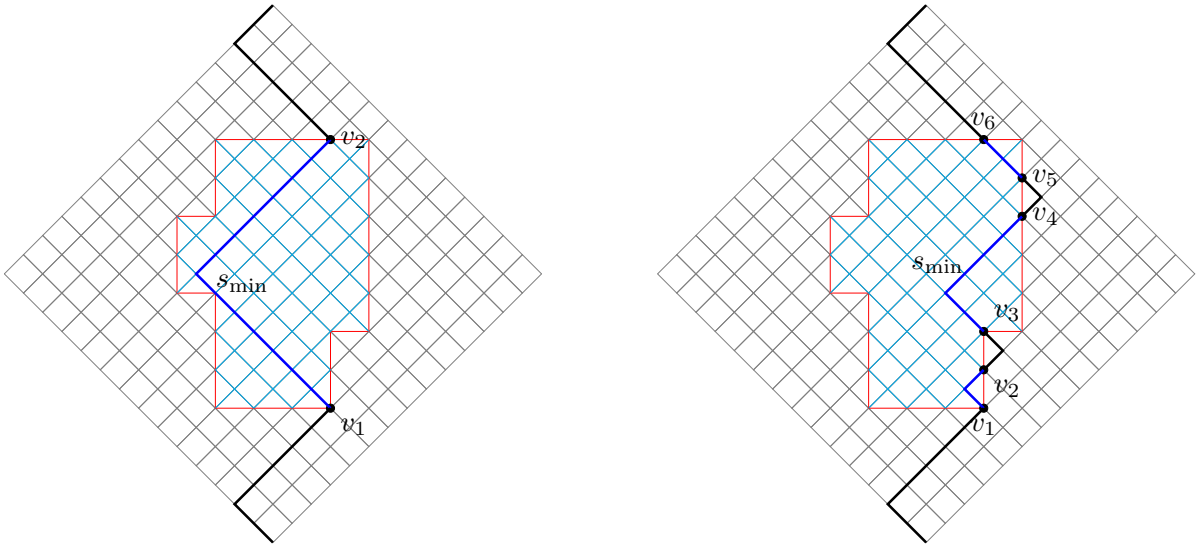

Figure 3.9: The minimal string $s^V_{\min}$ (black and blue line) corresponding to the boundary vertex set $V = \{v_1, v_2\}$ (left) and $V = \{v_1, ..., v_6\}$ (right). Within region A this minimal string is the initial string of the sub circuit. In $\bar{A}$ it is given by the backwards causal cones of the boundary vertices. Overall, the minimal string $s^V_{\min}$ represents the minimal in time evolved string that is compatible with the boundary vertex set V.

to the major circuit, for each sub circuit we can define an initial string from which all other string segments are obtained by a unitary evolution. In addition, we define the minimal string $s^V_{\min}$ to be the minimal evolved string corresponding to the boundary vertex set V. It corresponds to a partially completed circuit resulting in a definite qubit state $|\xi_{s^V_{\min}}\rangle$. Inside region A, the minimal string is the initial string of the sub circuit, outside of A it is given by the backwards causal cones of the boundary vertices. Examples of a minimal string $s^V_{\min}$ are displayed in figure 3.9. Furthermore, we define $\rho_{s^V_{\min}} \equiv \mathrm{Tr}_{q|s_{\bar{A}}} |\xi_{s^V_{\min}}\rangle \langle\xi_{s^V_{\min}}|$ to be the reduced density matrix of the qubit registers in the minimal string configuration. Thus, $\rho_{s^V_{\min}}$ is the density matrix for the qubits, which are present in the region A.
We claim that the reduced density matrix $\rho_{s^V_{\min}}$ of just the minimal string configuration differs only by a transformation with a unitary operator W from the second term in equation 3.45:

$$W^\dagger \left(\frac{1}{N(s^V_A)} \sum_{s^V_A, s'^V_A} \rho^V_{s_A, s'_A} \otimes |s^V_A\rangle \langle s'^V_A| \right) W = \rho_{s^V_{\min}} \otimes \frac{1}{N(s^V_A)} \sum_{s^V_A, s'^V_A} |s^V_A\rangle \langle s'^V_A| \tag{3.46}$$

The unitary operator W is similar to the one defined in section 1.2, but only acts within the sub circuit spanned by the strings s^V_A:

$$W = \sum_{s^V_A} V(s^V_{\min} \to s^V_A) \otimes |s^V_A\rangle \langle s^V_A| \tag{3.47}$$

$V(s^V_{\min} \to s^V_A)$ is the unitary operation acting on the data register that maps the minimal string configuration on the one corresponding to s^V_A. Equation 3.46 can be proved shortly:

$$\begin{aligned} & W^\dagger \left(\frac{1}{N(s^V_A)} \sum_{s^V_A, {s'}^V_A} \overbrace{V(s^V_{\min} \to s^V_A) \rho_{s^V_{\min}} V^\dagger(s^V_{\min} \to {s'}^V_A)}^{\rho^V_{s_A, s'_A}} \otimes |s^V_A\rangle \langle {s'}^V_A| \right) W \\ & = \frac{1}{N(s^V_A)} \sum_{s^V_A, {s'}^V_A} \underbrace{V^\dagger(s^V_{\min} \to s^V_A) V(s^V_{\min} \to s^V_A)}_{\mathbb{I}} \rho_{s^V_{\min}} \underbrace{V^\dagger(s^V_{\min} \to {s'}^V_A) V(s^V_{\min} \to {s'}^V_A)}_{\mathbb{I}} \\ & \qquad \otimes |s^V_A\rangle \langle {s'}^V_A| \\ & = \rho_{s^V_{\min}} \otimes \frac{1}{N(s^V_A)} \sum_{s^V_A, {s'}^V_A} |s^V_A\rangle \langle {s'}^V_A| \end{aligned} \tag{3.48}$$

Therefore, equation 3.45 can be rewritten as:

$$\begin{aligned} \rho_A &= \text{prob}\,(V = \emptyset)\, |\text{vac}\rangle \langle \text{vac}| + \sum_{V \neq \emptyset} \text{prob}(V)\, \underbrace{W \left(\rho_{s^V_{\min}} \otimes \frac{1}{N(s^V_A)} \sum_{s^V_A, {s'}^V_A} |s^V_A\rangle \langle {s'}^V_A| \right) W^\dagger}_{\rho^V} \\ &= \sum_V \text{prob}(V) \rho^V \end{aligned} \tag{3.49}$$

In the case $V = \emptyset$, the density matrix ρ^V is the vacuum state: $\rho^V = |\text{vac}\rangle \langle \text{vac}|$.
All states ρ^V corresponding to different boundary vertex sets V have support on orthogonal subspaces (two strings s^{V_1} and s^{V_2} cannot be equal, if $V_1 \neq V_2$). The vacuum state is also clearly orthogonal to all other states ρ^V. Using proposition 3.9 the entropy $S(\rho_A) = S(\sum_V \text{prob}(V)\rho^V)$ can be expressed by:

$$S(\rho_A) = H(\text{prob}(V)) + \sum_V \text{prob}(V) S(\rho^V) \tag{3.50}$$

Thus, the calculation of the entropy of a region A reduces to the calculation of the classical Shannon entropy $H(\text{prob}(V))$ of the probability distribution $\text{prob}(V)$, and the averaged von Neumann entropy of the individual density matrices ρ^V. The Shannon entropy does only depend on the geometry of the region, while the von Neumann entropy depends on the internal information and the entanglement of the circuit states. The entropy $S(|\text{vac}\rangle \langle \text{vac}|)$ of the vacuum state is zero and does not contribute to $S(\rho^A)$.
Density matrices that differ only by a unitary transformation lead to the same entropy, so we

can omit W and $W^\dagger$ in equation 3.49. Also by proposition 3.8, the entropy of a tensor product is additive. Hence, it follows:

$$S(\rho^V) = S(\rho_{s^V_{\min}}) + S\left(\frac{1}{N(s^V_A)} \sum_{s^V_A, s'^V_A} |s^V_A\rangle \langle s'^V_A|\right) \tag{3.51}$$

The entropy of the last term is zero, since its argument is a pure state (entanglement does only occur in the qubit register). We are left with an expression of $S(\rho^V)$ that is simple to evaluate:

$$S(\rho^V) = S(\rho_{s^V_{\min}}) = S\left(\mathrm{Tr}_{q|s_{\bar{A}}} |\xi_{s^V_{\min}}\rangle \langle \xi_{s^V_{\min}}|\right) \tag{3.52}$$

By proposition 3.13 this entropy is upper bounded by the number of qubits that are traced out. In turn this number is given by the length $l^V_{s_{\bar{A}}}$ of all string segments of $s^V_{\min}$ which lie outside region A. In the two examples of figure 3.9, $l^V_{s_{\bar{A}}}$ is obtained by the length of the black lines.

$$S(\rho^V) \leq l^V_{s_{\bar{A}}} \tag{3.53}$$

Finally, the probabilities prob(V) have to be calculated by combinatorics. However, for large and arbitrary shaped regions, this calculation will become very complex.
Notice that the derivation in this section did not depend on the circuit dimension, so it can likewise be applied on circuits involving two or more spatial dimensions.

3.3.3 Example

Consider the circuit in figure 3.10. Region A contains a single qubit at multiple times and is again encircled by the red line. Hence, we can regard region A as a one dimensional region in time which involves l different time steps. The circuit in this example is expected to have an interaction regime S (encircled by green lines), in which the non trivial gates act on the qubits. Outside this regime the internal qubit states are not changed and therefore each square represents the identity gate. In the following we calculate how the entanglement entropy of region A scales with its length l.
Beyond the interaction regime no entanglement is created. Thus when upper bounding the von Neumann entropies $S(\rho^V)$, it suffices to take only the qubits within S into account. In this example the possible boundary vertex sets V are either empty (strings lie completely outside of A), or involve exactly two vertices v_1 and v_2. For the region A with length l, there are l non empty boundary vertex sets V. Because we choose the interaction regime S to have a rectangular shape as in figure 3.10, the number n^{Tr} of qubits that are traced out, are equal for every reduced density matrix ρ^V $(V \neq \emptyset)$. Hence, each entropy is upper bounded by the same constant, which

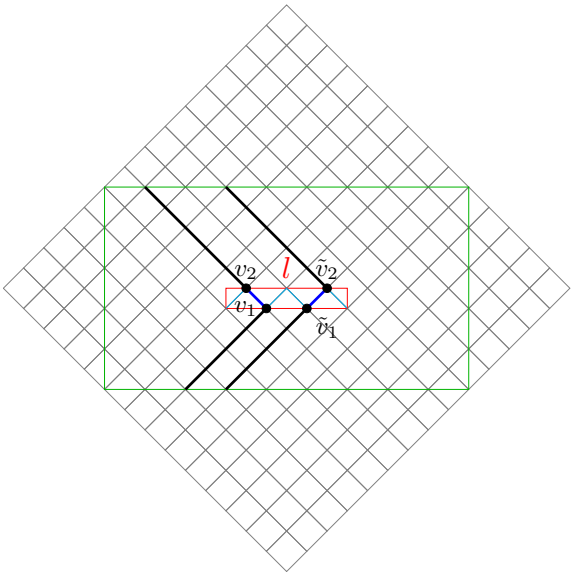

Figure 3.10: Region A (bounded by the red lines) contains one single qubit at multiple times. The number of time steps, it involves, is denoted by l, thus in this example: $l = 6$. The green rectangle displays the interaction regime S. Outside of S only identity gates are allowed and hence no entanglement is created. Two of the l possible boundary vertex sets V are shown ($V = \{v_1, v_2\}$ and $\tilde{V} = \{\tilde{v}_1, \tilde{v}_2\}$). The entropies $S(\rho^V)$ are upper bounded by the length of the black lines, which are independent of V and l. In this example the entropy $S(A)$ of region A scales with the length l.

is given by the length of the black lines in figure 3.10. In this case constant implies independence of the length l, since we are interested in varying only the length of the region.

$$\begin{aligned} S(\rho^V) &= n^{\mathrm{Tr}} = \text{const.} \qquad \text{if } V \neq \emptyset \\ S(\rho^V) &= 0 \qquad\qquad\quad \text{if } V = \emptyset \end{aligned} \tag{3.54}$$

In the following I assume that region A is centered within the circuit and its length l is small compared to the circuit size ($l \ll n$). In this way the probabilities $\mathrm{prob}(V \neq \emptyset)$ are approximately equal for all nonempty sets V. Therefore, the averaged von Neumann entropy is obtained by:

$$\sum_V \mathrm{prob}(V) S(\rho^V) \sim \mathrm{prob}(V \neq \emptyset) n^{\mathrm{Tr}} l = \mathcal{O}(l) \tag{3.55}$$

It scales with the length l of the considered region.
Let us calculate the probability $\mathrm{prob}(V \neq \emptyset)$. The total number N of strings in the circuit is:

$$N = \binom{2n}{n} \tag{3.56}$$

The number $N(V)$ of strings that are fixed to go through the boundary of region A is approximately given by:

$$N(V) \sim \binom{n}{\frac{n}{2}} \binom{n-1}{\frac{n}{2}} = \frac{1}{2} \binom{n}{\frac{n}{2}}^2 \tag{3.57}$$

Again, this approximation is good only for centered regions with $l \ll n$. From the last two equations it follows:

$$p \equiv \text{prob}(V \neq \emptyset) \sim \frac{1}{2}\binom{n}{\frac{n}{2}}^2 \Big/ \binom{2n}{n} = \text{const.} \tag{3.58}$$

The probability $\text{prob}(V = \emptyset)$ for a string not crossing region A is simply: $\text{prob}(V = \emptyset) \sim 1 - lp$. Then the Shannon entropy $H(\text{prob}(V))$ can be calculated by:

$$H(\text{prob}(V)) \sim -lp \log_2(p) - (1 - lp) \log_2(1 - lp) \tag{3.59}$$

Since p is constant, the first term scales as $\mathcal{O}(l)$. The second term can be upper bounded by using $-x \log_2 x \leq (1 - x)/\ln 2$ with $x = 1 - lp$:

$$H(\text{prob}(V)) \lesssim \mathcal{O}(l) + \frac{lp}{\ln 2} = \mathcal{O}(l) \tag{3.60}$$

Both, Shannon entropy and the averaged von Neumann entropy scale at most with the length l of region A. There might be tighter upper bounds than the one used for equation 3.60. In addition, for non centered regions or for ones where $l \ll n$ does not hold, the scaling will be different in general. However, for this specific example we obtain:

$$S(A) = \mathcal{O}(l) \tag{3.61}$$

If we consider the same one dimensional region A in a two dimensional circuit (two spatial dimensions), the scaling of entropy would be the same as in equation 3.61. The only values that change are the probabilities $\text{prob}(V)$, but in our approximation these probabilities are independent of the length l.

3.4 Conclusion and Outlook

In the first part of this chapter we showed that the one dimensional MERA obeys an area law for entanglement entropy that is common for quantum critical systems. The unitary circuit in one and two dimensions displays the simple area law corresponding to non-critical systems.
In the second part of this chapter we derived an expression for the entanglement entropy of a two dimensional region in space and time in the unitary circuit. As the important result we see that the entropy is obtained by two contributions (equation 3.50). One is given by the Shannon entropy of the probability distribution prob(V) and is pure geometric. The second contribution is the averaged von Neumann entropy of the individual density matrices ρ^V weighted by the probabilities prob(V). Hence, the latter contribution arises from the entanglement of the qubits.
Next, the developed tools can be applied to specific regions in the circuit as in the given example. The results can then be compared to the known area laws. Of substantial interest are regions large in time compared to space (or vice versa) and their higher dimensional analogs. Furthermore, we could consider the entanglement entropy between disconnected regions within the circuit and analyze the change in entropy when these regions move relative to each other.
In addition, the same procedure of section 3.3 can be applied to the MERA circuit. Fixing a specific region A, we can again find all possible boundary vertex sets and examine the minimal string configuration for each of them. If we consider the example of section 3.3.3 for a MERA circuit, we can no longer approximate the probabilities prob(V) to be equal, since the boundary vertex sets at the fine-grained part are expected to have a substantial higher probability than the ones at coarser scales. Therefore, the Shannon entropy and the averaged von Neumann entropy are likely to scale differently than in the unitary circuit.

Bibliography

[1] N. P. Breuckmann. From quantum circuits to hamiltonians: analysis of a multi-time construction for QMA. Master's thesis, RWTH Aachen, 2013.

[2] N. P. Breuckmann and B. M. Terhal. Space-time circuit-to-hamiltonian construction and its applications. *J. Phys. A: Math. Theor.*, 47:195304, 2014.

[3] G. Evenbly and G. Vidal. Algorithms for entanglement renormalization. *Phys. Rev. B*, 79:144108, 2009.

[4] G. Evenbly and G. Vidal. Tensor network states and geometry. *J. Stat. Phys.*, 145:891–918, 2011.

[5] G. Evenbly and G. Vidal. Quantum criticality with the multi-scale entanglement renormalization ansatz. *Strongly Correlated Systems, Numerical Methods*, Springer Series in Solid-State Sciences volume 176, Springer, 2013.

[6] G. Evenbly and G. Vidal. Scaling of entanglement entropy in the (branching) multi-scale entanglement renormalization ansatz. *Phys. Rev. B*, 89:235113, 2013.

[7] R. Feynman. Quantum mechanical computers. *Optic News*, 11:11–20, 1985.

[8] D. Gosset, B. M. Terhal, and A. Vershynina. Universal adiabatic quantum computation via the space-time circuit-to-hamiltonian construction. *Phys. Rev. Lett.*, 114:140501, 2015.

[9] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python. `http://www.scipy.org/`, 2001–. [Online; accessed 2015-07-28].

[10] A. Y. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and Quantum Computation.* Vol. 47 of Graduate Studies in Mathematics, American Mathematical Society, Providence, RI, 2002.

[11] A. Mizel, M. W. Mitchell, and M. L. Cohen. Energy barrier to decoherence. *Phys. Rev. A*, 63:040302, 2001.

[12] B. Mohar. The laplacian spectrum of graphs. *Graph Theory, Combinatorics, and Applications*, pages 871–898, Wiley, 1991.

[13] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, Boston, MA, USA, 2000.

[14] B. Swingle. Constructing holographic spacetimes using entanglement renormalization. arXiv:1209.3304, 2012.

[15] B. M. Terhal. Hand-written notes.

[16] G. van Rossum and F. L. Drake. Python Reference Manual. "`http://www.python.org`" Time Complexity: "`https://wiki.python.org/moin/TimeComplexity`". [Online; accessed 2015-06-23].

[17] E. Verlinde. On the origin of gravity and the laws of newton. *JHEP*, 1104, 2011:29, 2010.

[18] G. Vidal. A class of quantum many-body states that can be efficiently simulated. *Phys. Rev. Lett.*, 101:110501, 2008.

Appendix: Python Script

```
'''
This program generates all valid strings within a MERA, the Laplacian
 matrix associated to it, and its spectrum

:param n: Half the number of input qubits of the MERA
:param D: Time Depth of the MERA
'''

import numpy as np
import scipy.sparse as sps
from scipy.sparse.linalg import eigsh

def locationsOfSubstring(string, substring):
        '''
        Searches in the string for the occurrence of the substring 01

        :param string: original stringBinary
        :param substring: '01'

        :return locationsFound: Array containing all positions of
         substring in string
        '''
    substringLength = len(substring)
    def recurse(locationsFound, start):
        location = string.find(substring, start)
        if location != -1:
            return recurse(locationsFound + [location], location
             + substringLength)
        else:
            return locationsFound
    return recurse([], 0)

def insertIntoMatrix(stringDict, stringVector, matrixRows, matrixColumns,
 matrixIndex, matrixColumnIndex, stringVectorIndex, replacedString):
        '''
        The replaced string is inserted into stringVector, if it occurs
```

```
37          for the first time; The non-zero rows and columns of the
38           adjacency matrix are filled
39
40         :param stringDict: original stringDict
41         :param stringVector: original stringVector
42         :param matrixRows: original matrixRows
43         :param matrixColumns: original matrixColumns
44         :param matrixIndex: original matrixIndex
45         :param matrixColumnIndex: original matrixColumnIndex
46         :param stringVectorIndex: original matrixIndex
47         :param replacedString: original replacedString
48
49         :return matrixIndex + 1: original matrixIndex enlarged by one
50         :return matrixColumnIndex (+1): original matrixColumnIndex,
51          if string was generated earlier, it is unchanged
52         '''
53         # Search for occurrence of the replaced string in the dictionary
54         stringIndex = stringDict.get(int(replacedString, 2))
55         # If replaced string occurs for the first time append it
56         if stringIndex == None:
57                 stringVector.append(int(replacedString, 2))
58                 stringDict[int(replacedString, 2)] = len(stringVector) - 1
59                 matrixRows.append(stringVectorIndex)
60                 matrixColumns.append(matrixColumnIndex)
61                 return matrixIndex + 1, matrixColumnIndex + 1
62         else:
63                 matrixRows.append(stringVectorIndex)
64                 matrixColumns.append(stringIndex)
65                 return matrixIndex + 1, matrixColumnIndex
66
67 def calculateEvals(matrix, size):
68         '''
69         Calculates Eigenvalues and Eigenvectors of the Laplacian matrix
70
71         :param matrix: original laplacianMatrix
72         :param size: original stringVectorIndex; which now gives the
73          number of valid strings
74         '''
75         # Implicitly Restarted Arnoldi Method
76         # k: number of eigenvalues; which='SM': smallest eigenvalues
77         eigenvalues, eigenvectors = sps.linalg.eigsh(matrix, k = 5,
78          which='SM')
79
80         # Save
```

```
81          np.savetxt("Eigenvalues.txt", eigenvalues, fmt="%-8f")
82          np.savetxt("Eigenvectors.txt", eigenvectors, fmt='%+.8f')
83
84  def buildLaplacianMatrix(adjacencyMatrix, size):
85          '''
86          The degree matrix and Laplacian matrix are generated
87
88          :param adjacencyMatrix: original adjacencyMatrix
89          :param size: original stringVectorIndex; which now gives the
90           number of valid strings
91
92          :return laplacianMatrix
93          '''
94          # Array containing series 0,...,size; needed for degree matrix
95          degreeMatrixRows = np.arange(size)
96          # Entries of diagonal degree matrix
97          degreeMatrixValues = np.zeros(size)
98          for i in xrange (0, size):
99                  degreeMatrixValues[i] = adjacencyMatrix.getcol(i).nnz
100
101         # Degree Matrix: Diagonal elements of Laplacian matrix
102         degreeMatrix = sps.coo_matrix((degreeMatrixValues,
103         (degreeMatrixRows, degreeMatrixRows)), shape= (size,size))
104         laplacianMatrix = degreeMatrix - adjacencyMatrix
105
106         # Save
107         np.savetxt("laplacianMatrix.txt", laplacianMatrix.todense(),
108          fmt='%-5i')
109         return laplacianMatrix
110
111 def main():
112         #input
113         n = 2 # Half the number of input qubits of the MERA
114         D = 2 # Time depth of the MERA
115
116         # Length of the longest string
117         l_max = 3*(2**((D - n)/2 + 2))*((2**(n/2 - 1)) - 1) + 6 + 2*(D - n)
118
119         # Array of valid strings; first string given by initial string
120         stringVector = [int("0"*n + "1"*n, 2)]
121         # Dictionary containing the valid strings and their positions in
122          # stringVector as key
123         stringDict = {stringVector[0] : 0}
124
```

```
125         matrixRows = [] # non-zero rows of adjacency matrix
126         matrixColumns = [] # non-zero columns of adjacency matrix
127 
128         matrixIndex = 0 # Index of matrixRows and matrixColumns
129         matrixColumnIndex = 1 # An extra index for matrixColumns
130         stringVectorIndex = 0 # Index for stringVector
131 
132         while stringVectorIndex < len(stringVector):
133                 # Convert string into its binary representation
134                 stringBinary = bin(stringVector[stringVectorIndex])[2:]
135                 diff = stringBinary.count('1') - stringBinary.count('0')
136                 # Insert 0s at beginning if necessary
137                 stringBinary = "0"*diff + stringBinary
138                 # Search for sub-string 01 in current stringBinary
139                 location = locationsOfSubstring(stringBinary, '01')
140                 for position in xrange(0, len(location)):
141                         index = location[position]
142                         # Sub-string at even position
143                         if (index + 1) % 2 == 0:
144                             # Map string segment to 1010
145                                 replacedString = stringBinary[:index]
146                                  + "1010" + stringBinary[index+2:]
147                                 # Only generate strings up to length l_max
148                                 if len(replacedString) <= l_max:
149                                         matrixIndex, matrixColumnIndex
150                                          = insertIntoMatrix(stringDict,
151                                          stringVector, matrixRows,
152                                          matrixColumns, matrixIndex,
153                                          matrixColumnIndex,
154                                          stringVectorIndex, replacedString)
155                         # Sub-string at odd position
156                         else:
157                             # Map string segment to 10
158                                 replacedString = stringBinary[:index]
159                                  + "10" + stringBinary[index+2:]
160                                 matrixIndex, matrixColumnIndex
161                                  = insertIntoMatrix(stringDict,
162                                  stringVector, matrixRows,
163                                  matrixColumns, matrixIndex,
164                                  matrixColumnIndex, stringVectorIndex,
165                                  replacedString)
166                 stringVectorIndex += 1
167 
168         # Convert to NumPy arrays
```

```
169         npMatrixRows = np.array(matrixRows, dtype = np.uint64)
170         npMatrixColumns = np.array(matrixColumns, dtype = np.uint64)
171         npMatrixOnes = np.ones(matrixIndex, dtype = bool)
172         npStringVector = np.array(stringVector, dtype = np.uint64)
173 
174         # Upper diagonal of the adjacency matrix
175         adjacencyMatrix = sps.coo_matrix((npMatrixOnes,
176          (npMatrixRows,npMatrixColumns)), shape =
177          (stringVectorIndex,stringVectorIndex))
178         # Symmetric adjacency matrix
179         adjacencyMatrix = adjacencyMatrix + adjacencyMatrix.transpose()
180         laplacianMatrix = buildLaplacianMatrix(adjacencyMatrix,
181          stringVectorIndex)
182         calculateEvals(laplacianMatrix, stringVectorIndex)
183 
184         # Save
185         np.savetxt("stringList.txt", npStringVector, fmt="%-5i")
186         np.savetxt("MatrixRows.txt", npMatrixRows, fmt="%-5i")
187         np.savetxt("MatrixColumns.txt", npMatrixColumns, fmt="%-5i")
188         np.savetxt("adjacencyMatrix.txt", adjacencyMatrix.todense(),
189          fmt='%-5i')
190 
191 main()
```