TECHNISCHE UNIVERSITÄT WIEN

ATOMINSTITUT

DIPLOMA THESIS

# High Dimensional Quantum Computing: Qudit Circuit Compression for the Quantum Fourier Transform

for obtaining the academic degree

**Diplom-Ingenieur**

in the course of studies

**Master programme Technical Physics**

submitted by

**Pascal Windhager, BSc**

prepared at the
Institute of Atomic and Subatomic Physics
Faculty of Physics
TU Wien

supervised by
Univ.-Prof. Dr. Marcus Huber
Dr. Paul Erker

Vienna, 14.05.2025

(Signature Author) (Signature Supervisor)

TECHNISCHE UNIVERSITÄT WIEN

ATOMINSTITUT

DIPLOMARBEIT

# Hochdimensionales Quantencomputing: Qudit Gatterkomprimierung für die Quantenfouriertransformation

zur Erlangung des akademischen Grades

**Diplom-Ingenieur**

im Rahmen des Studiums

**Masterstudium Technische Physik**

eingereicht von

**Pascal Windhager, BSc**

ausgeführt am Atominstitut
der Fakultät für Physik
der Technischen Universität Wien

Betreuung
Univ.-Prof. Dr. Marcus Huber
Dr. Paul Erker

Wien, 14.05.2025

(Unterschrift Verfasser:in) (Unterschrift Betreuer:in)

# Abstract

The bottleneck of gate based quantum computation are entangling gates, which require precise control of the interaction between two or more quantum systems. Qubit based quantum computers are based on two-dimensional quantum systems and tend to use only a fraction of the available Hilbert space, while qudit based quantum computers harness unused degrees of freedom. The qudit embedding reduces the number of two-qubit gates by transforming them into either local qudit gates or into embedded two-qubit gates, which can subsequently be compressed into two-qudit gates using qudit circuit compression. In this thesis, the idea of qudit circuit compression is applied to the 4-qubit Simon's circuit and to the Quantum Fourier transform (QFT). By compressing multiple embedded two-qubit controlled rotation gates to two-qudit gates, it is possible to reduce the number of entangling gates significantly. One can show that the next-neighbour cut, that embeds adjacent qubits into qudits, is the best cut for the QFT, because it is always possible to reduce the number of two-qudit gates to one per qudit pair. Using a graph approach, where the nodes are qudits and the edges are (high-dimensional) controlled rotation gates, it is possible to deduce that the QFT resembles a fully-connected graph in the next-neighbour cut. Through this method it is possible to derive a formula for the number of required two-qudit gates as a function of the qudit dimension and the number of input qubits.

# Zusammenfassung

Die größte Herausforderung für gatterbasierte Quantencomputer sind verschränkende Gatter, da man dafür eine präzise Kontrolle über zwei oder mehr Quantensysteme benötigt. Quantencomputer die auf zwei-dimensionalen Quantensystemen basieren, tendieren dazu, nur einen Bruchteil des verfügbaren Hilbertraumes zu nutzen, wohingegen Qudit basierte Quantencomputer mehrere Freiheitsgrade nützen. Durch die Einbettung der Qubits in den Qudits reduziert sich die Anzahl an Zwei-Qubitgatter, da sie entweder zu lokale Gatter transformiert werden oder zu eingebettete Zwei-Qubitgatter, welche anschließend zu Zwei-Quditgatter komprimiert werden können mittels Qudit-Gatterkomprimierung. In dieser Arbeit wird die Idee der Qudit-Gatterkomprimierung auf den Simons Algorithmus für vier Qubits und auf die Quantenfouriertransformation (QFT) angewendet. Es wird gezeigt, dass der Nächste-Nachbar-Schnitt für die QFT, welcher benachbarte Qubits zu Qudits gruppiert, der bestmögliche Schnitt ist, da man dabei, mehrere verschränkende Gatter immer zu einem einzigen Zwei-Quditgatter pro Quditpaar, komprimieren kann. Mittels Graphentheorie, wobei die Qudits die Knoten und die Zwei-Quditgatter die Kanten darstellen, ist es möglich die QFT im Nächste-Nachbar-Schnitt immer als vollständig verbundenen Graphen darzustellen. Dadurch ist es möglich, eine Formel für die Anzahl an benötigten Zwei-Quditgatter, als Funktion der Quditdimension und der Anzahl an Eingangsqubits, herzuleiten.

# Statutory Declaration

I hereby declare that this thesis has been prepared in accordance with the code of conduct of TU Wien, rules for safeguarding good scientific practice, particularly without unauthorized assistance from third parties and without the use of any aids other than those explicitly indicated. Data and concepts directly or indirectly borrowed from other sources are properly cited.
This thesis has not been submitted, either in the same or a similar form, in any other examination procedure, either in Austria or abroad.

Pascal Windhager
Vienna, 14.05.2025

# Contents

# Chapter 1

# Introduction

Quantum computing is a disruptive technology that changes the way computation is envisioned. There are two reasons, why quantum computers are theoretically superior to classical computers. On the one hand, Moore's law cannot be continued for the next decades because classical chips are meanwhile at such a small scale that sooner or later quantum effects will play a crucial role, hence disturbing the classical computation. On the other hand, quantum computers can solve certain tasks, like integer factorization, exponentially faster than any known classical algorithm. The quantum speedup is based on either amplitude amplification, which is a generalization of Grover's algorithm that leads to a quadratic speedup, or on the Quantum Fourier transform (QFT) which leads to an exponential speedup, for problems that are part of the Abelian hidden subgroup, for example integer factorization, which plays a key role in cryptography because it is used for encryption algorithms like RSA. Quantum algorithms, as they are formulated today, are based on quantum bits, so called qubits, as the fundamental processing unit.[1]

Quantum computers cannot solve quantum algorithms at a practically relevant scale yet, because of noise and decoherence [2]. These limit the amount of entangling operations and consequently the circuit depth. As quantum systems tend to exploit only a fraction of the available Hilbert space, it is possible to use d-dimensional quantum system, called qudits, as the fundamental processing unit, in order to reduce the number of entangling gates. The process of embedding qubits within higher dimensional quantum systems is called qudit circuit compression. Qudit based quantum computing does not only reduce the number of necessary entangling gates, but also offers richer coherence [3] and entanglement structures [4] and it is also possible to formulate high dimensional quantum error correction[5, 6]. As the fundamental bottleneck are entangling gates [2], this approach is reasonable, as it can significantly reduces the number of two-qubit gates by transforming them into either local qudit gates or into embedded two-qubit gates, which can subsequently be compressed to two-qudit gates. In this thesis, the

idea of qudit circuit compression will be applied to Simon's algorithm and the QFT.

The thesis is structured as follows. First, the necessary background on quantum theory, quantum computers, the quantum circuit model, the Simon's algorithm, the QFT and qudit circuit compression will be explained. Since there are different ways to embed the qubits within the qudits, it is necessary to investigate the so called different cuts, but as the number of possible cuts vastly increases by increasing the number of qubits, it is necessary to investigate them for simple circuits first. The way of transforming two-qubit gates into two-qudit gates, is called high-dimensional gate construction. The idea of qudit circuit compression will be applied to the 4-qubit Simon's circuit in order to understand the concepts on a simple level. Subsequently, the QFT will be examined in detail in order to deduce the best possible cut for the QFT. In the end, a formula for the number of entangling gates for the QFT, embedded in any type of qudits, will be derived as a function of the qudit dimension and the number of input qubits.

# Chapter 2

# Theory background

In this chapter all the underlying theory for understanding chapter 3 is introduced. The first section 2.1 covers the profound difference between quantum theory and the classical world by introducing the postulates of quantum mechanics 2.1.1. Subsequently, concepts like quantum computers, which exploit the characteristics of quantum mechanics can be established. Quantum computers, which are covered in section 2.2, possess the capability of solving certain tasks, like integer factorization, exponentially faster than classical computers. These tasks are solved by quantum algorithms 2.4, which have a graphical representation called quantum circuit model 2.3. In standard literature like Nielsen [1], quantum algorithms are performed by encoding classical bits into quantum bits, called qubits, which is a system that can be represented by two quantum states, e.g. $|0\rangle$ and $|1\rangle$. Another way of executing quantum algorithms on a quantum computer is, to embed qubits into higher dimensional systems called qudits, which leads to a compressed quantum circuit, which is discussed in section 2.5. This so called qudit circuit compression of the Simon's algorithm and the Quantum Fourier transform is the main goal of this master thesis and is the topic of the next chapter 3.

## 2.1 Quantum theory

The starting point of quantum theory was set by Max Planck in 1900, formulating that the absorption or emission of energy can only happen in discrete energy packages, so called Quanta [7]. In the 1920s the mathematical framework was formalized, by introducing matrix mechanics and the wave formalism [8, 9]. Quantum physics differs profoundly from classical physics, which can be understood when looking at the postulates of quantum theory, which is covered in the next section 2.1.1.

### 2.1.1 Postulates of quantum theory

The postulates of quantum mechanics are a series of axioms or rules, which provide a formal framework for quantum theory. The postulates are formulated in various ways and are divided into either four or five, but in the following, five postulates will be formulated based on Dorobantu [10].

**First postulate**

> A physical system is completely described by its state vector $|\psi\rangle$, which is a unit vector in the system's state space, called Hilbert space $\mathcal{H}$.

The Hilbert space $\mathcal{H} \cong \mathbb{C}^d$ is a complex linear vector space of dimension $d$ equipped with a defined inner product. The quantum state $|\psi\rangle \in \mathcal{H}$ is represented in bra-ket-notation as a ket vector, introduced by Dirac [11]. The bra vector $\langle\psi| \in \mathcal{H}^*$ is a covector to $|\psi\rangle$, where $\mathcal{H}^*$ denotes the dual vector space. The bra vector $\langle\psi|$ can be interpreted as a linear functional that maps $|\psi\rangle$ to a number of the complex plane $\mathbb{C}$. The linearity of the Hilbert space implies that the a superposition of states is also a state of the system, if it is normalized to one.

**Second postulate**

> Every measurable physical observable X is described by an operator $\hat{X}$.

The operator associated to the observable is a Hermitian operator, which means that the matrix representing the operator is equal to the conjugate transpose of this matrix $H = H^\dagger = \overline{H^T}$. The eigenvectors of the operator $|x_i\rangle$ given by $\hat{X}\,|x_i\rangle = x_i\,|x_i\rangle$ form a complete basis of $\mathcal{H}$.

**Third postulate**

> All possible outcomes, of measuring a physical quantity X of a quantum system, are given by the eigenvalues $x_i$ of the corresponding operator $\hat{X}$. The probability of each outcome is given by the Born rule $p(x_i) = |\langle x_i|\psi\rangle|^2$.

One implication of this postulate is, that global phase factors do not matter in contrast to relative phases. The probabilities of the measurement outcomes of $\left|\psi'\right\rangle = e^{i\varphi}\,|\psi\rangle$ are

equal to those of $|\psi\rangle$ because $|e^{i\varphi}|^2 = 1$.

**Fourth postulate**

> The measurement process itself influences the quantum system. By projecting the quantum state $|\psi\rangle$ onto the operator $\hat{X}$, the state immediately becomes $|x_i\rangle$ corresponding to the measured eigenvalue $x_i$.

By preparing and measuring the quantum system multiple times a probability distribution can be obtained $\mathcal{P}(x_i)$. The expectation value of the operator is then given by (2.1).

$$\langle \hat{X} \rangle_\psi = \sum_i x_i \mathcal{P}(x_i) = \langle \psi | \hat{X} | \psi \rangle \tag{2.1}$$

Two observables can be measured simultaneously if and only if the commutator of the operators commute, which means $[\hat{X}, \hat{Y}] = \hat{X}\hat{Y} - \hat{Y}\hat{X} = 0$ [1].

**Fifth postulate**

> Every quantum system evolves according to the Schrödinger equation:
>
> $$i\hbar \frac{d}{dt} |\psi(t)\rangle = \hat{H}(t) |\psi(t)\rangle \tag{2.2}$$

$\hat{H}(t)$ denotes the Hamiltonian, which is an operator that corresponds to the total energy of the system. Given a quantum state $|\psi(t)\rangle$ at time $t$, the quantum state at time $t+\Delta t$ is given by $|\psi(t+\Delta t)\rangle = U(t+\Delta t, t) |\psi(t)\rangle$ where the unitary matrix $U$ is generated by the Hamiltonian of the system $U(t) = e^{-i\hat{H}(t)t/\hbar}$ with the reduced Planck constant $\hbar = h/(2\pi) = 6.626\,070\,15 \times 10^{-34}\,\mathrm{JHz}^{-1}/(2\pi)$. A matrix is unitary if its inverse is equal to its conjugate transpose, i.e. $U^{-1} = U^\dagger$.

### 2.1.2 Superposition, composite systems and entanglement

The Schrödinger equation is a linear differential equation, which implies that any linear combination of solutions of the Schrödinger equation is also a solution. The state of a quantum system, governed by the Schrödinger equation, is given by a linear combination, also called superposition, of the eigenvectors.

Composite systems are systems which consist of at least two subsystems. For example, consider two systems A and B with Hilbert spaces $\mathcal{H}_A$ and $\mathcal{H}_B$, then the Hilbert

space of the composite system is given by the tensor product of the two Hilbert spaces $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$. The tensor product of two vector spaces is commutative, whereas the tensor product of two vectors is not commutative, which means $|x\rangle_A \otimes |y\rangle_B \neq |y\rangle_B \otimes |x\rangle_A$. This can be seen in equation (2.3), where two two-dimensional vectors are tensored, which yields a four-dimensional vector. In the context of quantum computation, these two-dimensional vectors are called qubits, four-dimensional vectors ququarts and vectors with an arbitrary dimension $d$ qudits.

$$|z\rangle_{AB} = |x\rangle_A \otimes |y\rangle_B = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \otimes \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 y_0 \\ x_0 y_1 \\ x_1 y_0 \\ x_1 y_1 \end{pmatrix} \tag{2.3}$$

The tensor product of two matrices is not commutative as well, which can be seen in equation (2.4), where two $2 \times 2$ dimensional matrices are tensored to one $4 \times 4$ dimensional matrix. In the context of quantum computation a $2 \times 2$ dimensional matrix is called a local qubit gate and a $4 \times 4$ dimensional matrix a two-qubit gate or a local ququart gate.

$$Z_{AB} = X_A \otimes Y_B = \begin{pmatrix} x_{00} & x_{01} \\ x_{10} & x_{11} \end{pmatrix} \otimes \begin{pmatrix} y_{00} & y_{01} \\ y_{10} & y_{11} \end{pmatrix} = \begin{pmatrix} x_{00}y_{00} & x_{00}y_{01} & x_{01}y_{00} & x_{01}y_{01} \\ x_{00}y_{10} & x_{00}y_{11} & x_{01}y_{10} & x_{01}y_{11} \\ x_{10}y_{00} & x_{10}y_{01} & x_{11}y_{00} & x_{11}y_{01} \\ x_{10}y_{10} & x_{10}y_{11} & x_{11}y_{10} & x_{11}y_{11} \end{pmatrix} \tag{2.4}$$

A composite state which can be written in the form $|\psi\rangle_A \otimes |\psi\rangle_B$ is called separable or a product state. Any state which cannot be written in this separable form is called entangled. Entanglement is a crucial resource in quantum computation, since in most quantum algorithms the speedup is caused by entanglement [12], however, there are quantum algorithms which do not rely on entanglement [13, 14].

## 2.2 Quantum computers

A classical computer performs an algorithm by executing logic gates on the input bits. A bit is the smallest possible storage unit on a computer and takes the value of either 0 or 1. A logic gate takes one or more input bits, performs an operation and yields one or more output bits. For example, a NOT gate operates on just a single bit, by inverting the input bit, while an AND gate operates on two bits and the output is only 1 if both input bits are 1. These calculations or algorithms can be performed not only on bits, but also on multi-valued logic like trits, which take the value 0, 1 or 2. There exists no algorithm which can only be performed on multi-valued logic systems, but not on binary systems, which is proven by the Church-Turing thesis [15].

The Church-Turing thesis implies that all Turing-complete systems do have the same computational power. That means that any classical algorithm can be performed on a Turing-machine independent of the arithmetic of the machine [16]. The Turing-machine is a theoretical computational model, which defines an abstract machine that can simulate any algorithmic process by a set of rules [17]. Since there is no obvious reason that binary systems are superior to other systems, the first supercomputers in the 1940s were based on decimal arithmetic [18]. Binary arithmetic is simpler and more compact in terms of hardware implementation, therefore, the arithmetic speed is increased [19].

A quantum computer is equivalent to a classical computer in a sense that it also performs algorithms by executing gates on the input quantum bits (qubits) [1]. A qubit is a two-level quantum system, which can be represented in the computational basis as the quantum states $|0\rangle$ and $|1\rangle$, analogous to a classical bit, but the state can also be in any superposition of these states, according to the postulates 2.1.1.

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \;\text{ with } \{\alpha, \beta\} \in \mathbb{C} \tag{2.5}$$

In equation (2.5), $\alpha$ and $\beta$ represent the probability amplitudes. Upon measurement, the state $|\psi\rangle$ collapses either into $|0\rangle$ or $|1\rangle$ with the probability of $|\alpha|^2$ or $|\beta|^2$ respectively, according to the Born rule. Due to the normalization of the wavefunction $|\alpha|^2 + |\beta|^2 = 1$ holds.

Analogously to the classical computer, the quantum computer can be based not only on qubits, but also on higher-dimensional quantum systems, called qudits. For example, a four dimensional quantum system, called ququart, can encapsulate the information of two qubits.

A quantum algorithm is a sequence of quantum gates, which are given by unitary operators according to the fifth postulate of quantum mechanics. A gate which acts only on a single quantum system, for instance a qubit or ququart, is called a local gate because it acts only on the local Hilbert space of the quantum system. An entangling gate, also called two-qudit or multi-qudit gate, instead acts on two or more quantum systems, respectively and therefore entangles them. There are a lot of different types of quantum gates and a few of them will be covered in section 2.2.3.

There are two reasons why quantum computers can be advantageous to classical computers. On the one hand, classical computers are limited in scaling down, since quantum effects appear at tiny scales in the processing units, hence Moore's law cannot be continued in the next decades. On the other hand, quantum computers can solve certain tasks, like integer factorization, exponentially faster, but they can also simulate quantum systems efficiently, in contrast to classical computers [1]. The distinction between efficient and inefficient and the difference between problems that a quantum computer can solve efficiently compared to a classical computer, will be discussed in the next section 2.2.1.

### 2.2.1 Complexity classes

It is not proven but believed by many researchers that a classical computer cannot simulate a quantum system efficiently. As the number of complex numbers needed to describe the quantum system grows exponentially with the size of the system, the classical computation needs $c^n$ bits, whereas the quantum computation takes only $kn$ qubits, where $k$ and $c$ are constants determined by the simulated system.

A complexity class, groups problems that are described by the same kind of features. The different combinations of features give rise to a lot of complexity classes, whereby P and NP are the most important ones to describe the difference between classical and quantum computation. All problem classes which are in P can be solved efficiently, which means in polynomial time on a classical computer. However, NP describes all problem classes which are solvable in non-polynomial time on a classical computer. P is a subset of NP but as mentioned before, it is an unsolved problem if P and NP are the same complexity class. NP-complete is a subclass of NP, with the distinction that all NP-complete problem classes are equally hard, which means that an algorithm which solves an NP-complete problem can solve all other NP-complete problems. If P is not equal to NP, all NP-complete problems cannot be simulated efficiently on a classical computer.

Integer factorization is believed to be a problem of the type NP, since there exists no algorithm that can solve this problem efficiently on a classical computer. Shor's algorithm is a quantum algorithm, which solves the integer factorization problem, represented by $n$ bits, by using $3n$ qubits [20]. It is based on the Quantum Fourier transform (QFT) as many other quantum algorithms and is therefore an interesting candidate for qudit circuit compression. There is also another version of Shor's algorithm which uses only $2n + 3$ qubits [21].

### 2.2.2 Computational basis

The computational basis states for a qubit are given by $|0\rangle$ and $|1\rangle$ and they are usually represented as two-dimensional vectors, as shown in equation (2.6).

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{2.6}$$

The label 0 or 1 of the ket vector indicates the state of the quantum system. The computational basis states for two qubits are given by the tensor product of the qubits, which yields four possible states $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$. The computational basis states for $n$-qubits are given by $|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle$. The labels of these states, expressed in binary representation of a n-qubit system, correspond to a n-bitstring $\underline{x} \in \{0,1\}^{\otimes n}$, which is defined as $n$ bits concatenated in a sequence and is referenced by either $\underline{x} =$

$x_0x_1x_2\ldots x_{n-1}$ or $\underline{x} = (x_0, x_1, x_2, \ldots, x_{n-1})$. The value of x in decimal arithmetic can be recovered through $\underline{x}$ with the binary representation $x = x_0 2^{n-1} + x_1 2^{n-2} + \cdots + x_{n-1}2^0$, which implies that $x_0$ is the most significant bit and $x_{n-1}$ is the least significant bit. This representation can be used to express the binary fraction $0.x_l x_{l+1}\ldots x_m = x_l/2 + x_{l+1}/2^2 + x_m/2^{m-l+1}$, which is a useful shorthand notation to express the phases in the QFT.

The computational basis states for a ququart are $|0\rangle$, $|1\rangle$, $|2\rangle$ and $|3\rangle$, where these labels in binary representation are equal to the labels of the basis states for two qubits. Like in the qubit case, the basis states for two ququarts are given by the tensor product, hence 16 states are possible, which are given in binary representation as $|0000\rangle$, $|0001\rangle$, $|0010\rangle$, $|0011\rangle$, $|0100\rangle$, …, $|1111\rangle$. These 16 basis states are equal to the quhex basis states $|0\rangle$, $|1\rangle$, …, $|15\rangle$ if they are also expressed in binary notation.

Expressing the state in binary representation for any number of qudits of dimension $d$ is always possible and it can be used to construct higher dimensional gates, which is done in section 3.1.

### 2.2.3 Quantum gates

A quantum gate is a unitary matrix that acts on one (local gate) or more (entangling gate) qudits. Since the qubit is a two-level quantum system, a local qubit gate has a shape of a $2 \times 2$ matrix. A gate which entangles two qubits has consequently a shape of a $4 \times 4$ matrix. As two qubits can be encoded in one ququart, the two-qubit entangling gate becomes a local gate in the ququart. Therefore the local ququart gate has a shape of a $4 \times 4$ matrix and the two-ququart entangling gate has a shape of a $16 \times 16$ matrix. This two-ququart gate can then again be encoded in a local quhex (16-dimensional system, which encapsulates four qubits or two ququarts) gate and consequently the two-quhex entangling gate has a shape of a $256 \times 256$ matrix. This can be continued for systems of arbitrary dimension d, where $d = 2$, $d = 4$, $d = 8$ and $d = 16$ represent a qubit, ququart, quoct and a quhex, respectively. Consequently, the shape of the local qudit gate is $d \times d$ and the two-qudit gate has a shape of a $d^2 \times d^2$ matrix.

In the following, quantum gates which are used in the Simon's algorithm 2.4.1 and in the QFT 2.4.2 are discussed.

#### Identity gate

The identity gate is an identity matrix, therefore leaves the quantum state unchanged. The identity matrix for a qudit of dimension $d$ is denoted like $\mathbb{1}_d$. In the following sections, it is used for the construction of high-dimensional qudit gates, where some subsystems stay unchanged.

### Hadamard gate

The Hadamard gate performs the Hadamard transformation on a qudit and is a type of Fourier transform. The Hadamard gate $H$ for a single qubit is shown in equation (2.7) and it maps $|0\rangle$ and $|1\rangle$ to $(|0\rangle + |1\rangle)/\sqrt{2}$ and $(|0\rangle - |1\rangle)/\sqrt{2}$, respectively and vice versa.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2.7}$$

The Hadamard transformation can also be defined recursively which is useful if a $H$ gate is applied to each qubit within a qudit. The recursive formula is given in equation (2.8) and is equivalent to equation (2.7) for $n = 1$.

$$H^n = \frac{1}{\sqrt{2}} \begin{pmatrix} H^{n-1} & H^{n-1} \\ H^{n-1} & -H^{n-1} \end{pmatrix} \text{ with } H^0 = 1 \tag{2.8}$$

The recursive definition of the Hadamard transformation is only useful if an $H$ gate is applied to each qubit in the qudit. The QFT algorithm is constructed in such a way that the Hadamard gate is only applied to a single qubit, while the remaining qubits remain unaffected. Equation (2.9) presents a qudit Hadamard gate that acts on $N$ qubits within the qudit and operates non-trivially only when the labels $x_i$ are equal to one.

$$H_{\underline{x}} = \bigotimes_{i=0}^{N-1} H^{x_i} \tag{2.9}$$

For example, $H_{11}$ applies an $H$ gate to both qubits in the ququart and is equivalent to $H^2$ of equation (2.8). However, $H_{01} = \mathbb{1}_2 \otimes H$ cannot be described by equation (2.8), as it applies an $H$ gate to the second qubit within the ququart while leaving the first qubit unchanged.

Throughout this whole thesis, $H_n$ will be used to denote an operation which applies a Hadamard gate to the $n$-th qubit in the qudit and leaves all other qubits unchanged, shown in equation (2.10), and $H_{\underline{x}}$ will only be used for two qubits which yields the three gates $H_{10}$, $H_{01}$ and $H_{11}$. This notation proves particularly valuable in the calculations of the QFT presented in section 3.3.

$$H_n = \mathbb{1}_2^{\otimes n-1} \otimes H \otimes \mathbb{1}_2^{\otimes N-n} \tag{2.10}$$

### Rotation gate

The rotation gate, also called phase shift gate, modifies the phase of the quantum system. Applied on a qubit in the computational basis, it leaves the $|0\rangle$ state unchanged and maps $|1\rangle$ to $e^{i\varphi}|1\rangle$. Equation (2.11) shows the corresponding gate $R(\varphi)$, which performs this kind of operation, where $\varphi$ is the phase factor. $R_j$, shown in equation

(2.11), represents a special form of $R(\varphi)$, because it applies the phase $e^{2\pi i/2^j}$ to $|1\rangle$ and it is used in the QFT algorithm 2.4.2.

$$R(\varphi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix} \text{ or } R_j = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^j} \end{pmatrix} \tag{2.11}$$

**NOT gate**

The NOT gate inverts the input bits and it exists also on a classical computer. The quantum NOT gate does exactly the same, since it maps $|0\rangle$ and $|1\rangle$ to $|1\rangle$ and $|0\rangle$, respectively, for the computational basis. For a superposition, like it is given in equation (2.5), it interchanges the probability amplitudes $\alpha$ and $\beta$. The matrix representation of the quantum NOT gate is equivalent to the Pauli-x matrix $\sigma_x$ and is shown in equation (2.12).

$$NOT = \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.12}$$

**Controlled gates**

The Hadamard, rotation and NOT gate are all local gates, which means that they do not entangle the qubits, since they are only applied to a single qubit. A controlled gate is an entangling gate, because it changes the target qubit depending on the state of the control qubit. Any local gate can be controlled by a control qubit, therefore it is possible to define an arbitrary controlled unitary gate, shown in equation (2.13).

$$CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{pmatrix} \tag{2.13}$$

The controlled NOT gate (CNOT) and the controlled rotation gate (CR) are used in the Simon's algorithm and in the QFT, respectively. The CNOT and CR matrices can be obtained by replacing the unitary matrix in equation (2.13), by the corresponding matrices, given in the previous subsections.

### 2.2.4 Hardware implementation

Any kind of quantum system can act as a quantum computer as long as it fulfills certain criteria. These criteria can be summarized by DiVincenzo's criteria [22]:

1. The Hilbert space of a quantum system must grow exponentially fast with the system size. That means that the Hilbert space is made of the direct product of multiple smaller subsystems.

2. It must be possible to prepare the quantum system to a fiducial quantum state.

3. The quantum system needs to be isolated well enough, so that the error is low enough to be able to correct it.

4. It must be possible to perform controlled unitary operations between the quantum subsystems up to a small error.

5. The coupling to the measurement apparatus must be strong enough, so that the state of the wavefunction has a large impact on the result of the measurement.

The impossibility of completely isolating a quantum system from its environment, entangles the environment with the quantum system which leads to quantum decoherence [23]. There are two type of errors in quantum computers, namely bit-flip and phase-flip errors. Bit-flip errors have the same impact as a NOT gate, whereas phase-flip errors induce an unwanted rotation gate. Both of these errors are crucial, if uncorrected, the result of the quantum algorithm is incorrect. Since it is impossible to isolate a gate-based quantum computer completely from its environment, these errors need to be corrected. Quantum error correction deals with this problem, by performing quantum error correction codes like the 2D surface code. [24]

Any quantum system that fulfills DiVincenzo's criteria can perform quantum computation. There are different implementations of quantum systems, each with its own advantages and disadvantages regarding decoherence, state preparation, gate errors, measurement, etc. A non-exhaustive list of quantum computing architectures is superconducting, trapped ion, neutral atom, photonic, nuclear magnetic resonance and nitrogen-vacancy centred quantum computing [25]. Since quantum computing architectures are different, it is only natural that the gates which can be executed efficiently with a low error, which are also called native gates, differ across platforms [1]. The complexity of a quantum gate can be defined as the number of native gates required to implement the desired operation.

## 2.3 Quantum circuit model

Quantum circuits are a theoretical model to visually depict a quantum algorithm, consisting of an initial state, a sequence of quantum gates and measurements. The visual representation of the quantum circuit was first used by Feynman [26], which is a modification of the Penrose graphical notation [27].

The time in a quantum circuit diagram flows from left to right, which implies that the input qubits are on the left hand side and the measurement is on the right hand side of the diagram. The notation for quantum circuit diagrams for qubits is well established [28] and an exemplary circuit is presented in figure 2.1a. It shows a quantum algorithm,

which applies a Hadamard gate $H$, a CNOT gate, a controlled unitary gate $U_1$ and an arbitrary two-qubit gate $U_2$ sequentially to the input qubits $|x_0\rangle$ and $|x_1\rangle$. The last symbol in the quantum circuit 2.1a depicts the measurement process, which collapses the qubit into a classical bit.

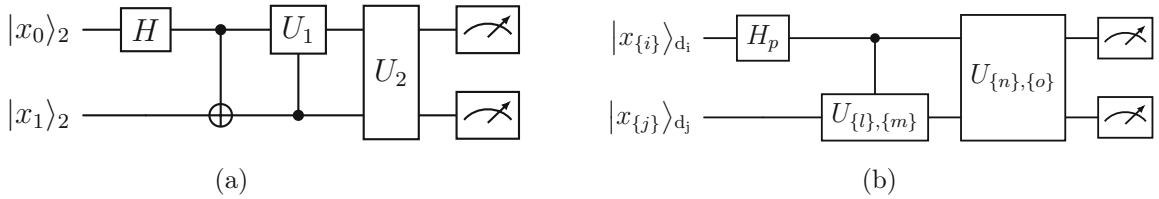



Figure 2.1: Example quantum circuits drawn with the `quantikz` [29] package. In (a), a two-qubit quantum circuit, which applies a Hadamard $H$, CNOT, controlled unitary $U_1$ and an arbitrary unitary $U_2$ gate sequentially, is shown. In (b), a two-qudit circuit with a high-dimensional Hadamard $H_p$, controlled unitary $U_{\{l\},\{m\}}$ and an arbitrary unitary $U_{\{n\},\{o\}}$ gate, is shown. The last symbol in both circuits represents the measurement of the qubit or qudit.

The quantum circuit 2.1b shows a two-qudit circuit, where each qudit $\left|x_{\{i\}}\right\rangle_{d_i}$, $\left|x_{\{j\}}\right\rangle_{d_j}$ is of dimension $d_i$, $d_j$ and embeds $\{i\}$, $\{j\}$ qubits, where $d_i = 2^{|\{i\}|}$, $d_j = 2^{|\{j\}|}$ and $|\{i\}|$, $|\{j\}|$ is the number of elements in the set. There is no common notation for qudits yet, therefore, the notation for qudit circuits will be explained in the following. The quantum circuit 2.1b shows an algorithm which applies a Hadamard gate $H_p$, a high-dimensional controlled unitary gate $U_{\{l\},\{m\}}$ with $\{l\}$ target qubits and $\{m\}$ control qubits, and an arbitrary unitary gate $U_{\{n\},\{o\}}$, with no defined control or target qubits or one qudit acts as both a control and a target qudit. The high-dimensional Hadamard gate $H_p$ applies a qubit Hadamard gate to the qubit $p$ within the qudit and applies an identity gate to all remaining qubits.

It is possible to reverse the order of two adjacent gates, if they commute with each other. As a consequence it is possible to compress adjacent two-qudit gates to a single two-qudit gate. If there are multiple local gates between two entangling gates, the commutators of all possible combinations of the local gates need to be checked, to determine whether the two entangling gates can be aligned next to each other. Diagonal matrices always commute with each other, which means that all commutators between (high-dimensional) controlled rotation gates are zero.

## 2.4 Quantum algorithms

As already mentioned, a quantum algorithm is a sequence of quantum gates applied on the input states. There are two main techniques, the QFT and amplitude amplification,

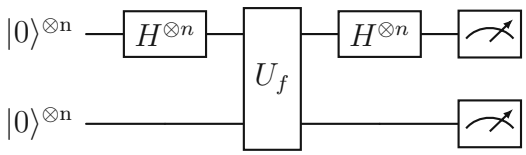



Figure 2.2: Quantum circuit for Simon's algorithm for 2n qubits. Before and and after the entangling unitary querying function gate, a Hadamard gate is applied to the first qubit register and in the end the quantum states are measured.

which are used by most quantum algorithms [1]. Amplitude amplification is a generalization of Grover's algorithm [30], which is a quantum search algorithm which leads to a quadratic speedup compared to the best known classical algorithm. Algorithms that use the QFT generally imply an exponential speedup. It is used in algorithms like the Deutsch-Jozsa algorithm [31], Bernstein-Vazirani algorithm [32], Simon's algorithm [33], Shor's algorithm [20], quantum phase estimation algorithm [1], or in general, for problems that are part of the Abelian hidden subgroup [1].

A quantum algorithm, which solves the Abelian hidden subgroup problem, encompasses all quantum algorithms which use the QFT. It finds the period of a given periodic function independent of the domain and range of the periodic function. [1]

### 2.4.1 Simon's algorithm

Simon's algorithm solves Simon's problem in exponentially fewer queries than the best classical algorithm. Simon's problem questions if the output of a function is invariant when applying an XOR mask to the input. XOR is the abbreviation for the exclusive OR gate, which yields 0 if both inputs are equal, which means it yields 1 if one input is 0 while the other is 1. The XOR gate can also be understood as the bitwise addition of two binary numbers modulo 2. In the example (2.14) and (2.15), the XOR operation is applied to two bit-strings of length three.

$$s_1 = 010 \quad s_2 = 110 \quad \text{with } s_i \in \{0,1\}^3 \tag{2.14}$$

$$s_1 \oplus s_2 = (0,1,0) \oplus (1,1,0) = (0+1 \bmod 2, 1+1 \bmod 2, 0+0 \bmod 2) = 100 \tag{2.15}$$

The definition of Simon's problem is, given a function $f : \{0,1\}^n \to \{0,1\}^n$ that maps a $n$-bit-string to a $n$-bit-string, which fulfills the condition $f(x \oplus s) = f(x)$, find the hidden bit-string $s$ by querying $f$. Any classical algorithm would need exponential oracle queries to solve this problem, while Simon's algorithm only needs $\mathcal{O}(n)$ queries [33]. Simon's algorithm consists of two steps, namely querying the function $f$ $n$-times by executing the quantum circuit 2.2 and classical post-processing that identifies the

secret string $s$. The quantum circuit 2.2 consists of two quantum registers, where each register contains $n$ qubits. First, a Hadamard gate is applied to the first register, then the unitary gate which encapsulates the querying function is applied to both registers and before measuring, another Hadamard gate is applied to the first register. The measurement of the first register yields a string $z \in \{0,1\}^{\otimes n}$, which fulfills condition (2.16). [33]

$$z\dot{s} = 0 \tag{2.16}$$

Running the circuit $m = \mathcal{O}(n)$ times, yields $z_m$ bit strings and by solving this system of linear equations via e.g. Gaussian elimination [34] the hidden string $s$ can be found.

### 2.4.2 Fourier transform

The Fourier transform $\mathcal{F}(\omega)$ of a time-dependent signal or function $f(t)$ is given by equation (2.17), which transforms the domain of the function from time to frequency. If the function is localized in time, $\mathcal{F}(\omega)$ will be spread across the frequency domain and vice versa. The inverse Fourier transform is equivalent to equation (2.17) up to a sign change in the exponential function, the integral domain changes from time to frequency and the functions $f(t)$ and $\mathcal{F}(\omega)$ are interchanged.

$$\mathcal{F}(\omega) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i t\omega} dt \tag{2.17}$$

In signal processing, an analog signal can only be measured at certain time steps, therefore the function is discretized. The Fourier transform of such a signal is the discrete Fourier transform (DFT), shown in equation (2.18). Due to the discretization, the integral becomes a sum, where $N$ is the number of samples and $f[0], f[1], \ldots, f[N-1]$ represent the values of the signal at each time step. The inverse to the DFT is equivalent to (2.18) up to a sign in the exponent and that the discrete functions $\mathcal{F}[\omega]$ and $f[t]$ are interchanged again [35]. The Fourier transform in general is not restricted to the time and frequency domain, but can be applied to any complementary variables, like position and momentum.

$$\mathcal{F}[\omega] = \frac{1}{\sqrt{N}} \sum_{t=0}^{N-1} f[t] e^{-2\pi i t\omega/N} \tag{2.18}$$

The discrete Fourier transform (DFT) on $n$ bits, denoted $\mathrm{DFT}_n$, operates on $2^n$ entries and is computed by naive algorithms in $\mathcal{O}((2^n)^2)$ steps. The best classical algorithm for calculating the DFT is the fast Fourier transform (FFT) which calculates the $\mathrm{DFT}_n$ in $\mathcal{O}(n2^n)$ steps, which is still exponential in time [12]. The DFT can also be expressed as a single matrix, shown in equation (2.19), where $\omega = e^{-2\pi i/N}$ and $N = 2^n$. By adding

another bit to the DFT matrix, its size is doubled which means it grows exponentially in size.

$$\mathrm{DFT}_n = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \dots & \omega^{(N-1)} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \dots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \dots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{(N-1)} & \omega^{2(N-1)} & \omega^{3(N-1)} & \dots & \omega^{(N-1)(N-1)} \end{pmatrix} \text{ with } N = 2^n \quad (2.19)$$

The quantum Fourier transform (QFT) is the quantum analogue to the DFT. With a quantum computer it is possible to calculate the $\mathrm{QFT}_n$ within $\mathcal{O}(n^2)$ steps which is polynomial in time [12]. The QFT maps a basis state $|0\rangle, |1\rangle, \dots, |x\rangle, \dots, |N-1\rangle$ to a superposition of all basis states, multiplied by the respective amplitudes, as shown in equation (2.20).

$$\mathrm{QFT}_n |x\rangle = \sum_{k=0}^{N-1} e^{-2\pi i x k/N} |k\rangle \quad (2.20)$$

The product representation, shown in equation (2.21), is equivalent to equation (2.20) which is proven in appendix A [1].

$$\mathrm{QFT}_n |x_0 x_1 \dots x_{n-1}\rangle =$$

$$\frac{(|0\rangle + e^{-2\pi i 0.x_{n-1}} |1\rangle) \otimes (|0\rangle + e^{-2\pi i 0.x_{n-2} x_{n-1}} |1\rangle) \otimes \dots \otimes (|0\rangle + e^{-2\pi i 0.x_0 x_1 \dots x_{n-1}} |1\rangle)}{2^{n/2}} \quad (2.21)$$

This product representation is suitable to construct the QFT quantum circuit, which is shown in figure 2.3. First, a Hadamard gate is applied to the qubit $|x_0\rangle$ which transforms the quantum state $|x_0 \dots x_{n-1}\rangle$ to the quantum state shown in equation (2.22) because $e^{-2\pi i 0.x_0} = 1$ if $x_0 = 1$ and $e^{-2\pi i 0.x_0} = -1$ if $x_0 = 0$. Second, a controlled rotation gate $R_2$ is applied to the state, which yields equation (2.23). For all remaining qubits, controlled rotation gates $R_3, \dots, R_{n-1}$ are applied to the first qubit $|x_0\rangle$, which yields the quantum state shown in equation (2.24). This state is exactly the same state as the last qubit state in equation (2.21). Therefore, the outermost qubits are swapped in the end in order to obtain the desired state. If the QFT is at the end of a quantum algorithm these SWAP gates can be omitted and replaced by classical post-processing. The aforementioned procedure can be repeated for the remaining qubits, with the modification that there are $i$ fewer controlled rotation gates per qubit, where $i$ is the index of the input qubit. The resulting quantum state is equivalent to the state

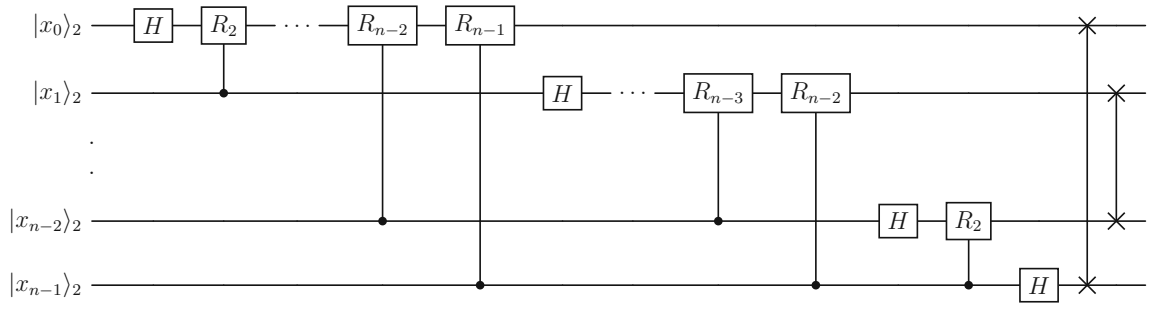



Figure 2.3: Quantum circuit for the QFT for $n$ qubits, which consists of Hadamard gates $H$, controlled rotation gates $R_i$ and SWAP gates. The quantum circuit can be derived by the product representation of the QFT, shown in equation (2.21).

shown in equation (2.21), up to the SWAP gates that exchange the first and the last qubit, the second with the second-last qubit, and so on.

$$|\psi\rangle = \frac{1}{2^{1/2}}\left(|0\rangle + e^{-2\pi i 0.x_0}|1\rangle\right)|x_1 \dots x_{n-1}\rangle \tag{2.22}$$

$$\rightarrow \frac{1}{2^{1/2}}\left(|0\rangle + e^{-2\pi i 0.x_0 x_1}|1\rangle\right)|x_1 \dots x_{n-1}\rangle \tag{2.23}$$

$$\rightarrow \frac{1}{2^{1/2}}\left(|0\rangle + e^{-2\pi i 0.x_0 x_1 \dots x_{n-1}}|1\rangle\right)|x_1 \dots x_{n-1}\rangle \tag{2.24}$$

As stated before, the QFT is executable in $\mathcal{O}(n^2)$ steps, which can be seen in the quantum circuit 2.3, which requires $n$ Hadamard gates, $n(n-1)/2$ controlled rotation gates and $n/2$ SWAP gates, where each SWAP gate can be constructed with three CNOT gates [1].

The derivation above features the QFT in a qubit embedding, but in the following chapter the qubits are embedded in qudits. Three qubits can be encoded in one quoct but as the QFT is discussed only for even number of qubits the quoct-encoding of the $\mathrm{QFT}_3$, $\mathrm{QFT}_6$, $\mathrm{QFT}_9$, …, $\mathrm{QFT}_{3n}$ will not be investigated in detail. A comparison between quocts and ququarts can be made by examining the $\mathrm{QFT}_6$ embedded in two quocts and in three ququarts, as presented in section 3.3.3. To compare quocts with quhexes one could investigate the $\mathrm{QFT}_{12}$ embedded in four quocts as opposed to three quhexes.

## 2.5 Qudit circuit compression

Quantum computing based on qudits provides several advantages over qubit based quantum computers, as they offer richer coherence [3] and entanglement structures [4], but also improved quantum error correction [5, 6]. The qubits are encoded in higher

dimensional quantum systems, called qudits, in order to reduce the non-local gate count. There are different ways to group the qubits, therefore it is necessary to find the best so called cut to reduce the entangling gate count to the lowest possible number.

For example, a quantum circuit that consists of four qubits $|x_0\rangle_2 |x_1\rangle_2 |x_2\rangle_2 |x_3\rangle_2$ will be compressed to two ququarts, but there are three different ways to group these qubits. The next-neighbour cut $|x_{01}\rangle_4 |x_{23}\rangle_4$ groups the qubits which are next to each other, the inner-outermost cut $|x_{03}\rangle_4 |x_{12}\rangle_4$ groups the two outermost qubits and the two innermost ones and the next-next-neighbour cut $|x_{02}\rangle_4 |x_{13}\rangle_4$ groups the qubits 0 and 2 and the qubits 1 and 3. These two ququarts can then be embedded in a quhex, where all gates of the quantum circuit are reduced to local gates. The number of possible cuts vastly increases, when increasing the number of qubits.

By embedding the qubits into qudits, the resulting gates are still two-qubit gates embedded in a higher dimensional Hilbert space, therefore they are called embedded two-qubit gates. However, these gates are harder to perform than a regular two-qubit gate because they are often not natively available. Therefore, genuine qudit-entangling gates, which do not admit a tensor-product structure, can be developed by exploiting the high dimensional Hilbert space. [36]

The possibility to create completely new gates admits the possibility to create new quantum algorithms, which are based on genuine qudit entangling gates. In general, the number of entangling gates is reduced even further by exploiting these qudit-entangling gates. For example, a four-qubit quantum circuit with multiple two-qubit gates can be embedded in two ququarts, which yields multiple embedded two-qubit gates. These gates, which represent $16 \times 16$ matrices, can then be multiplied with each other to obtain a two-ququart gate. If there are local gates $U_L$ between the entangling gates, these can be extended by the identity $U_{L,emb} = U_L \otimes \mathbb{1}_4$, which yields an embedded local ququart gate $U_{L,emb}$, which can then be incorporated in an already existing two-ququart gate. This might increase the complexity of the two-ququart gate, but as the goal of qudit circuit compression is to minimize the number of entangling gates while increasing the complexity of the genuine qudit entangling gates as little as possible, this method might not be the best. One way to circumvent this problem is, to rearrange the order of the gates if they commute. Subsequently, adjacent two-qudit gates can be compressed to a single two-qudit gate.

In the next chapter 3, the idea of qudit circuit compression is applied to the 4-qubit Simon's circuit and to the QFT.

# Chapter 3

# Qudit circuit compression

In chapter 2, all the underlying theory, in order to understand the following chapter, is discussed. By embedding the qubits into qudits the two-qubit gates are transformed into either local ququart gates or embedded two-qubit gates. The construction of these high dimensional gates is explained in section 3.1. In the next section 3.2, the quantum circuit for Simon's algorithm for four qubits is discussed. The number of entangling gates of this circuit is reduced, by exploiting the methods of qudit circuit compression. As there is no general formulation of the Simon's circuit for more than four qubits, it is not possible to investigate the properties of the qudit Simon's circuit further. Therefore, a more useful algorithm, the QFT, is discussed in detail. First, the simplest cases of the QFT are investigated, to set the foundation for the generalization to arbitrary number of qubits. In the final section 3.3.6, a formula for the number of two-qudit gates as a function of the input qubits $N$ and the qudit dimension $d$, is derived.

## 3.1 High dimensional gate construction

In this section, the high dimensional gate construction formalism is discussed only for the CNOT and controlled rotation gate CR, because these are used by the Simon's algorithm and the QFT, respectively. In order to embed the two-qubit gate into the higher dimensional Hilbert space, it is useful to express the dimension $d$ in binary representation. The two affected qubits in the binary representation are exposed to the desired effect, like performing a CNOT gate or applying a phase. In the following, the gate construction of the two-qubit gates CNOT and CR embedded in two ququarts, is shown explicitly.

The computational basis states of a two-ququart system are given in section 2.2.2. The two qubits within the two ququarts, which are affected by the controlled unitary gate, are the indices of interest in the $16 \times 16$ matrix. As an example, two ququarts $|x_{01}\rangle$ and $|x_{23}\rangle$ are examined in the next-neighbour cut. The CNOT gate between the

first and the third qubit, which are the control and target qubits respectively, becomes an embedded two-qubit gate. The relevant states are those with a trailing one, as the CNOT gate acts non-trivially only when the control qubit is in the state $|1\rangle$. Therefore, the states $|1000\rangle$, $|1001\rangle$, $|1100\rangle$, $|1101\rangle$ are mapped to the states $|1010\rangle$, $|1011\rangle$, $|1110\rangle$, $|1111\rangle$ respectively, and vice versa or expressed in set notation $\{|x\rangle : x \in \{0,1\}^4 \mid x_0 = 1, x_2 \to x_2 \oplus 1\}$, where $1 \oplus 1 = 0$. The described matrix is spanned by figure 3.1, where the bit strings in the top row and the left column are shown for better identification of the desired states. The states in the upper left corner, where $x_0 = 0$, stay invariant because of the identity matrix $\mathbb{1}_8$. The two $4 \times 4$ matrices in the lower right corner, which map the states to $x_2 \to x_2 \oplus 1$, represent the base matrix X, shown in equation (3.1), which is used in the $C_{02}$ and $C_{12}$ gates in the 4-qubit Simon's circuit in the upcoming section (3.2).

| | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000 | 1 | | | | | | | | | | | | | | | |
| 0001 | | 1 | | | | | | | | | | | | | | |
| 0010 | | | 1 | | | | | | | | | | | | | |
| 0011 | | | | 1 | | | | | | | | | | | | |
| 0100 | | | | | 1 | | | | | | | | | | | |
| 0101 | | | | | | 1 | | | | | | | | | | |
| 0110 | | | | | | | 1 | | | | | | | | | |
| 0111 | | | | | | | | 1 | | | | | | | | |
| 1000 | | | | | | | | | 0 | 0 | 1 | 0 | | | | |
| 1001 | | | | | | | | | 0 | 0 | 0 | 1 | | | | |
| 1010 | | | | | | | | | 1 | 0 | 0 | 0 | | | | |
| 1011 | | | | | | | | | 0 | 1 | 0 | 0 | | | | |
| 1100 | | | | | | | | | | | | | 0 | 0 | 1 | 0 |
| 1101 | | | | | | | | | | | | | 0 | 0 | 0 | 1 |
| 1110 | | | | | | | | | | | | | 1 | 0 | 0 | 0 |
| 1111 | | | | | | | | | | | | | 0 | 1 | 0 | 0 |

Figure 3.1: High-dimensional gate construction of the embedded two-qubit CNOT gate between the first and third qubit, as the control and target qubit, respectively, embedded in two ququarts in the next-neighbour cut. All states, where the first qubit is zero $\{|x\rangle : x \in \{0,1\}^4 \mid x_0 = 0\}$, remain identical, therefore they are multiplied with the identity matrix $\mathbb{1}_8$, as shown in the upper left corner. For the lower right corner, the first qubit is one, therefore, all states are mapped to its corresponding state $\{|x\rangle : x \in \{0,1\}^4 \mid x_0 = 1, x_2 \to x_2 \oplus 1\}$. The $4 \times 4$ matrices represent the base matrix X, shown in equation (3.1), which is used for the $C_{02}$ and $C_{12}$ gates in the next-neighbour cut of the 4-qubits Simon's circuit.

In the next-neighbour cut the qubits are adjacent, therefore the standarad binary

representation works. In order to construct high-dimensional gates in other cuts than the next-neighbour cut, the indices need to swapped according to the cut structure. For four qubits $|x_0x_1x_2x_3\rangle$ there are only two more cuts, namely the inner-outermost and the next-next-neighbour cut, where the qubits are mapped according to $\{|x\rangle : x \in \{0,1\}^4 \mid x_2 \to x_3, x_3 \to x_4, x_4 \to x_2\}$ and $\{|x\rangle : x \in \{0,1\}^4 \mid x_2 \to x_3, x_3 \to x_2\}$, respectively. As a consequence, the values of the binary representation in figure 3.1 need to be rearranged, hence the high dimensional gates need to be constructed according to the new indices.

The controlled rotation gate, which is used in the QFT, is a diagonal matrix, therefore it is only necessary to consider a vector, labeled by the computational basis states in binary representation. Only if both control and target qubit are one, the gate acts non-trivially. For example, a controlled rotation gate CR between the first and second qubit would yield a local ququart gate in the next-neighbour cut. The first and second qubit are only one in the last four states of the 16 basis states, which yields the gate $R_{01} = \mathrm{CR} \otimes \mathbb{1}_4$. An example for an embedded two-qubit gate would be a CR gate between any qubit of the first ququart and second ququart, which can be constructed in the exact same way. For example, the matrix construction of the embedded two-qubit gate $R_{02}$ between the first and third qubit in the next-neighbour cut is shown in figure 3.2, where $\omega_i$ needs to be replaced with $\omega_3$ for the $R_{02}$ gate. There, all states where the first and third qubit are one, are multiplied by $\omega_i$, all other states remain identical, which can also be expressed as $\{|x\rangle : x \in \{0,1\}^4 \mid x_0 = 1, x_2 = 1, |x\rangle \to \omega_i |x\rangle\}$. The two 4-dimensional vectors $(1, 1, \omega_i, \omega_i)$ represent the diagonal of the base matrix $A_i$, shown in equation (3.8), which is used in the QFT.

This gate construction formalism can be applied to any multi-qudit gate, therefore, it is used throughout the entire thesis.

$$\mathrm{diag}\Big(\begin{array}{c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c} 0000 & 0001 & 0010 & 0011 & 0100 & 0101 & 0110 & 0111 & 1000 & 1001 & 1010 & 1011 & 1100 & 1101 & 1110 & 1111 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \omega_i & \omega_i & 1 & 1 & \omega_i & \omega_i \end{array}\Big)$$

Figure 3.2: High-dimensional gate construction of the embedded two-qubit CR gate between the first and third qubit, as the control and target qubit, respectively, embedded in two ququarts in the next-neighbour cut. All states, where the first qubit is zero $\{x \in \{0,1\}^4 \mid x_0 = 0\}$, remain identical and are therefore multiplied with a one for all eight entries, as shown in the left-hand part. For the right-hand side, the first qubit is one, therefore, all states, where the third qubit is also one, are multiplied by $\omega_i$ or expressed more compactly $\{x \in \{0,1\}^4 \mid x_0 = 1, x_2 = 1, x \to \omega_i x\}$. The two 4-dimensional vectors represent the diagonal of the base matrix $A_i$, shown in equation (3.8), which is used for the QFT in general.

## 3.2 4-qubit Simon's algorithm

The details of Simon's algorithm are discussed in section 2.4.1, where the general formulation of the Simon's circuit is given, which consists of a Hadamard gate on the first qubit register before and after applying the querying function gate to both quantum registers. There is no general formulation of the querying function gate, but for four qubits it is known, that it can be implemented with four CNOT gates, shown in the quantum circuit 3.3. In figure 3.4, a graph approach is used to illustrate the aforementioned quantum circuit, where the four qubits (blue nodes $|0_i\rangle$) are embedded in two ququarts (red rectangles) in three different ways. With this graph approach it is directly visible, which gates become local ququart gates and those which remain entangling gates, i.e. embedded two-qubit gates. There, the next-neighbour cut seems to be the worst cut, as it does not reduce the number of entangling gates, but by combining the gates to a two-ququart entangling gate, it is possible to reduce the number of entangling gates to one. These three different cuts, shown in figure 3.4, are discussed in more detail in the following sections.

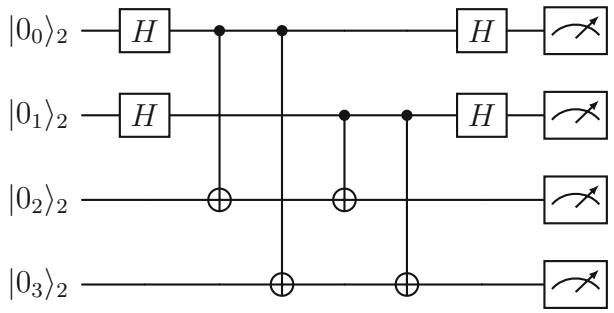
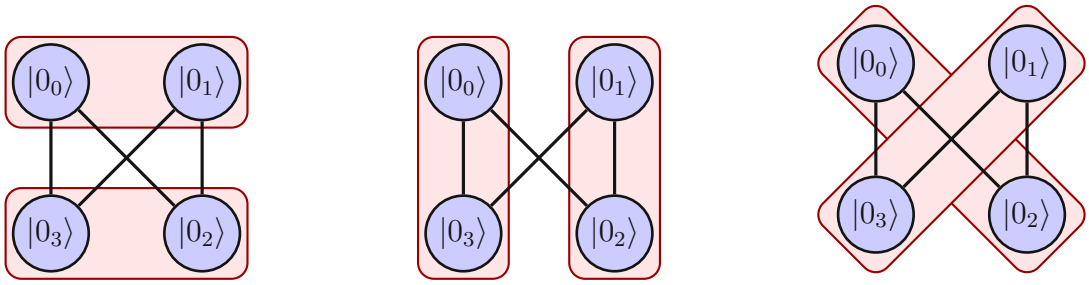



Figure 3.3: Quantum circuit for Simon's algorithm for four qubits. Hadamard gates $H$ are applied to the first register like in the general Simon's circuit 2.2 and the querying function is implemented with four CNOT gates.

### 3.2.1 Next-neighbour cut

The next-neighbour cut embeds the qubits 0 and 1 and the remaining two qubits 2 and 3 each in a ququart. The three base matrices $X, Y$ and $Z$, shown in equation (3.1), are built out of the tensor product of the Pauli-X matrix $\sigma_x$ and the identity matrix. These base matrices are $4 \times 4$ matrices, therefore they can either be viewed as entangling




(a) Next-neighbour cut (b) Inner-outermost cut (c) Next-next-neighbour cut

Figure 3.4: All possible cuts of Simon's circuit for four qubits represented with graphs. In each cut, four qubits (blue nodes $|0_i\rangle$) are embedded in two ququarts (red rectangles). The edges represent CNOT gates, which manifest as embedded two-qubit gates $C_{ij}$ in the ququart embedding. Through the cut specific embedding, without compressing adjacent gates, cut (a) does not reduce the number of entangling gates, while (b) and (c) reduce four CNOT gates to two local ququart gates and two embedded two-qubit gates.

two-qubit gates or local ququart gates.

$$X = \sigma_x \otimes \mathbb{1}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad Y = \mathbb{1}_2 \otimes \sigma_x = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$
$$Z = \sigma_x \otimes \sigma_x = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \tag{3.1}$$

The quantum circuit for the next-neighbour cut is shown in figure 3.5, where the $H_{11} = H \otimes H$ gate is a Hadamard gate applied to the qubits 0 an 1 and the $C_{ij}$ gates are controlled NOT gates, where the first index is the control qubit and the second one is the target qubit. All $C_{ij}$ gates become embedded two-qubit gates, because the control and target qubits are always in two different ququarts. These gates, shown in equation (3.2), are block matrices composed solely of the base matrices $X$ and $Y$ and the identity matrix $\mathbb{1}_4$.

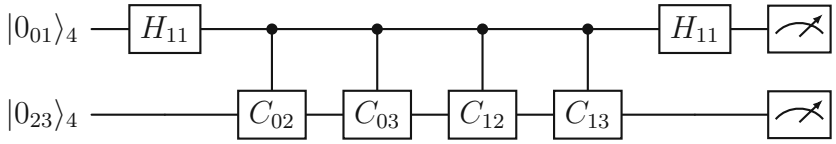



Figure 3.5: Next-neighbour cut for the four qubit Simon's circuit. The first two and the last two qubits are each embedded in one ququart.

$$C_{02} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & X & \\ & & & X \end{pmatrix} \quad C_{03} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & Y & \\ & & & Y \end{pmatrix}$$
$$C_{12} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & X & & \\ & & \mathbb{1}_4 & \\ & & & X \end{pmatrix} \quad C_{13} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & Y & & \\ & & \mathbb{1}_4 & \\ & & & Y \end{pmatrix} \tag{3.2}$$

The embedding of the ququarts in the next-neighbour cut does not reduce the entangling gates, because no two-qubit gate becomes a local ququart gate, due to aforementioned reasons. However, the entangling gates can be reduced to one, by multiplying all embedded two-qubit gates $C_{ij}$, which yields a two-ququart gate which affects all four qubits embedded in the two ququarts. The resulting quantum circuit is shown in figure 3.6.

$$C_{01,23} = C_{13}C_{12}C_{03}C_{02} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & Z & & \\ & & Z & \\ & & & \mathbb{1}_4 \end{pmatrix} \tag{3.3}$$

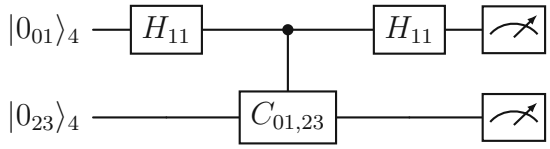



Figure 3.6: Compressed quantum circuit for the next-neighbour cut. Four embedded two-qubit gates $C_{ij}$ are compressed to one qudit entangling gate $C_{01,23}$ or more specifically a two-ququart gate.

### 3.2.2 Inner-outermost cut

The embedding of the qubits in the inner-outermost cut yields two ququarts, which embed the outermost qubits 0 and 3 and the innermost ones 1 and 2. Therefore, the $C_{03}$ and $C_{12}$ gates become local ququart gates, as the indices already suggest, because the control and the target qubit are in the same ququart. The $C_{03}(C_{12})$ gate, shown in equation (3.4), is composed of a local ququart CNOT gate that is applied to the first (second) ququart, tensored with the identity matrix $\mathbb{1}_4$, which leaves the second (first) ququart invariant. The $C_{02}$ gate is equivalent to the $C_{03}$ gate of the next-neighbour cut and the $C_{13}$ gate is built of up and down matrices $U$ and $D$, which are shown in equation (3.5).

The local ququart gates $C_{03}$ and $C_{12}$, given in equation (3.4), are $16 \times 16$ matrices that act on the entire Hilbert space while the gates shown in the quantum circuit 3.7, which are also referenced by $C_{03}$ and $C_{12}$, act only on the respective ququart. This ambiguity that $C_{03}$ is used as a $16 \times 16$ matrix in the calculations of commutators but also as a $4 \times 4$ local ququart gate in the quantum circuit, is used throughout the whole thesis. This simplifies the expressions for the commutators but also allows one to reference the gates $C_{03}$ and $C_{12}$ as two different gates although both are a CNOT gate applied to its respective ququart.

$$C_{02} = C_{03\_NN-Cut} \qquad C_{03} = \text{CNOT} \otimes \mathbb{1}_4 \qquad C_{12} = \mathbb{1}_4 \otimes \text{CNOT} \tag{3.4}$$

$$C_{13} = \begin{pmatrix} U & D & & \\ D & U & & \\ & & U & D \\ & & D & U \end{pmatrix} \qquad U = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 0 & \\ & & & 0 \end{pmatrix} \qquad D = \begin{pmatrix} 0 & & & \\ & 0 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \tag{3.5}$$

The quantum circuit 3.7 consists of these four controlled NOT gates $C_{ij}$ and of $H_{10} = H \otimes \mathbb{1}_2$ gates, which apply a Hadamard gate to the first qubit and leave the second qubit unchanged inside the ququart.

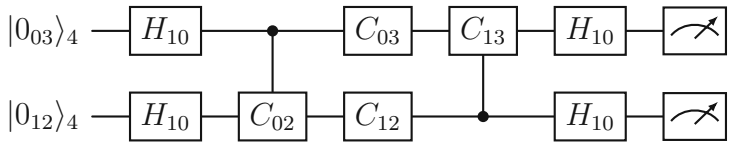



Figure 3.7: Inner-outermost cut for the four qubit Simon's circuit. The first and the last qubit and the second and third qubit are each embedded in a ququart.

The first gate $C_{02}$ in the quantum circuit 3.7 can be switched with both local ququart gates because the commutators $[C_{02}, C_{03}]$ and $[C_{02}, C_{12}]$ are zero. It would also be

possible to switch the local ququart gates with the last embedded two-qubit gate $C_{13}$ because the commutators between those gates are also zero. The embedded two-qubit gates $C_{02}$ and $C_{13}$ can then be combined to a two-ququart entangling gate $C^{ent}_{01,23}$, which yields the compressed quantum circuit 3.8. This entangling gate, shown in equation (3.6), is composed of the already introduced $U$ and $D$ matrices, but also of the matrices $U_H = HUH^{-1}$ and $D_H = HDH^{-1}$.

$$C^{ent}_{01,23} = \begin{pmatrix} U & D_H & & \\ D_H & U & & \\ & & D & U_H \\ & & U_H & D \end{pmatrix} \qquad C^{ent\&loc}_{01,23} = \begin{pmatrix} U & D & & \\ D & U & & \\ & & U_H & D_H \\ & & D_H & U_H \end{pmatrix} \tag{3.6}$$

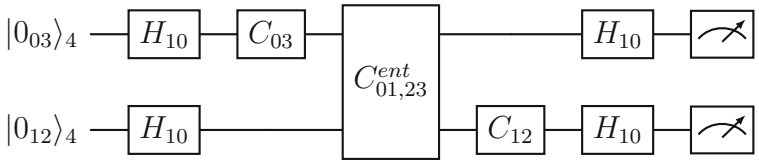



Figure 3.8: Compressed quantum circuit for the inner-outermost cut. Embedded two-qubit gates $C_{02}$ and $C_{13}$ commute with the local ququart gates $C_{03}$ and $C_{12}$, therefore the order can be reversed and the embedded two-qubit gates can be compressed to a single two-ququart gate $C^{ent}_{01,23}$.

The hardness of the gate depends on the quantum computing architecture, therefore it might be possible that the combination of all four controlled NOT gates is easier to implement. This yields a new gate $C^{ent\&loc}_{01,23} = C_{12}C^{ent}_{01,23}C_{03}$, shown in equation (3.6), which is constructed of the same base matrices $U$, $D$, $U_H$ and $D_H$ as the $C^{ent}_{01,23}$ gate, but in a different arrangement.

As the qubits were encoded in a different order, classical post-processing, i.e. bit permutation, is necessary, in order to receive the same results as in the next-neighbour cut. If Simon's circuit is a subroutine in a quantum algorithm, SWAP gates need to be applied, if the next subroutine is processed in a different cut, to ensure the correctness of further quantum computation.

### 3.2.3 Next-next-neighbour cut

In the next-next-neighbour cut the first and the third qubit and the second and the last qubit are each embedded in a ququart. Therefore, the $C_{02}$ and $C_{13}$ gates become local in the first and second ququart, respectively. This can be seen in the quantum circuit, shown in figure 3.9, where $H_{10}$ is the same Hadamard gate as in the inner-outermost cut and the four controlled NOT gates are exactly the same gates as in the

inner-outermost cut, but in a different order. The local ququart gates $C_{02} = \text{CNOT} \otimes \mathbb{1}_4$ and $C_{13} = \mathbb{1}_4 \otimes \text{CNOT}$ are built again of a CNOT gate and the identity matrix. The embedded two-qubit gates $C_{03}$ and $C_{12}$ are equal to the $C_{02}$ and $C_{13}$ gates of the inner-outermost cut, respectively. The $C_{03}$ gate is also equal to the $C_{03}$ gate of the first cut which means this gate is present in all three different cuts. These relations are also shown in equation (3.7).

$$C_{02} = C_{03\_IO-C} \quad C_{03} = C_{02\_IO-C} = C_{03\_NN-C} \quad C_{12} = C_{13\_IO-C} \quad C_{13} = C_{12\_IO-C} \tag{3.7}$$

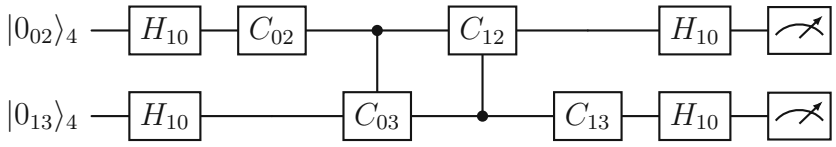



Figure 3.9: Next-next-neighbour cut for the four qubit Simon's circuit. The first and third qubit aswell as the second and fourth qubit are each embedded in a ququart.

The embedded two-qubit gates are already aligned next to each other so they can be combined again to a $C^{ent}_{01,23}$ gate which is equal to the $C^{ent\&loc}_{01,23}$ of the second cut. Combining the two local ququart gates and the two embedded two-qubit gates yields a controlled NOT gate that encompasses all four gates $C^{ent\&loc}_{01,23}$ which is equal to the $C^{ent}_{01,23}$ gate of the inner-outermost cut.

The number of entangling gates can be reduced to one for any cut so by performing this cut analysis no obvious advantage could be gained. The reason that the number of entangling gates can be reduced to one for any cut lies in the symmetry of Simon's circuit, which implicates that all relevant commutators are zero. The structure of the entangling gates in the first cut is different than in the other two cuts, therefore it is up to the quantum computing architecture which gates are easy to implement and therefore choose the best cut for this architecture. One other way to quantify the advantage of a certain cut is to compute the entanglement power of these controlled NOT gates, to compare which one has the lowest cost to create this kind of entanglement.

In the next section, the qudit circuit compression formalism is applied to the QFT, where it is shown that the number of two-qudit gates is cut-dependent, in contrast to the 4-qubit Simon's circuit.

## 3.3 Quantum Fourier Transform

The quantum Fourier transform is the quantum analogue to the discrete Fourier transform, covered in section 2.4.2. If the QFT is done on a single high-dimensional quantum

system with a single gate, this gate is exactly the DFT matrix. As the size of the DFT matrix increases exponentially by increasing the number of bits, this approach is not scalable.

In the following sections, the qudit circuit compression is applied to the simplest cases of the QFT on $2n$ qubits. In this process, the different cuts are discussed and in the last sections, the insights are generalized to any number of qudits and as a consequence a formula for the number of two-qudit gates is derived as a function of the input qubits $N$ and qudit dimension $d$.

As already discussed for Simon's circuit, classical post processing is necessary for all cuts, but the next-neighbour cut, to obtain the correct results. As the circuit for the $\mathrm{QFT}_N$, defined in figure 2.3, uses SWAP gates at the end, the cut-dependent classical post-processing can be combined with the SWAP-post-processing. If the QFT is a subroutine in a quantum algorithm, the corresponding SWAP gates, that match the cut of the next quantum subroutine, need to be applied to receive the correct results. For the rest of this chapter, the SWAP gates are typically omitted, as the precise structure depends on the use case and can often be handled classically.

### 3.3.1 2-qubit QFT

The circuit for the QFT for two qubits can be seen in figure 3.10. It consists of Hadamard gates $H$ and a controlled rotation gate $R_2$. The two qubits can be embedded in a single ququart, therefore, there exists only one possible cut. The first Hadamard gate $H$ becomes a $H_{10} = H \otimes \mathbb{1}_2$ gate, the controlled rotation gate, which is a two-qubit gate becomes a local ququart gate and the last Hadamard gate becomes a $H_{01} = \mathbb{1}_2 \otimes H$ gate. These three local ququart gates can be combined to a single local ququart gate, which is equivalent to the $\mathrm{DFT}_2$ matrix.

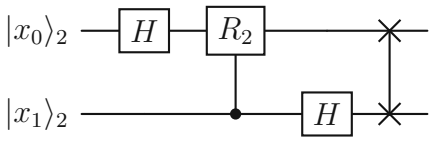
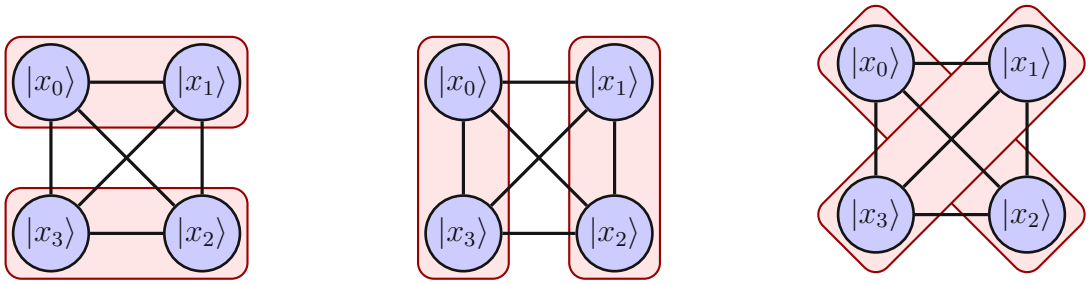



Figure 3.10: Quantum Fourier transform for two qubits

### 3.3.2 4-qubit QFT embedded in two ququarts

The QFT for four qubits is shown in the quantum circuit 3.12, which consists of Hadamard gates $H$ and controlled rotation gates $R_i$, where the index $i$ defines the phase, which is applied to the qubit as defined in equation (2.11). Similar to the previous section, four qubits are involved, therefore, there exist three different cuts. In figure

3.11, these three cuts are shown, where the blue nodes represent the qubits $|x_i\rangle$, the red rectangles the ququart embedding $|x_{ij}\rangle$ and the edges are the controlled rotation gates $R_i$. In all three cuts, the embedded two-qubit gates $R_{ij}$ are built of the base matrices $A_i$ and $B_i$, shown in equation (3.8), two-qubit controlled rotation gates $R_i$ and identity matrices. The $C_{ij}$ matrix, shown in equation (3.9), is the multiplication of the base matrices $A_i$ and $B_j$ and is used in the compressed controlled rotation gates. In the following subsections, the three different cuts, shown in figure 3.11, are discussed.




(a) Next-neighbour cut (b) Inner-outermost cut (c) Next-next-neighbour cut

Figure 3.11: All possible cuts of the $\mathrm{QFT}_4$, where each cut embeds four qubits (blue nodes $|x_i\rangle$) in two ququarts (red rectangles). The edges represent a controlled rotation gate $R_i$.

$$A_i = \begin{pmatrix} 1 & 0 \\ 0 & \omega_i \end{pmatrix} \otimes \mathbb{1}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \omega_i & 0 \\ 0 & 0 & 0 & \omega_i \end{pmatrix} \quad B_i = \mathbb{1}_2 \otimes \begin{pmatrix} 1 & 0 \\ 0 & \omega_i \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \omega_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \omega_i \end{pmatrix} \tag{3.8}$$

$$C_{ij} = A_i \cdot B_j \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \omega_j & 0 & 0 \\ 0 & 0 & \omega_i & 0 \\ 0 & 0 & 0 & \omega_i\omega_j \end{pmatrix} \tag{3.9}$$

### Next-neighbour cut

In the next-neighbour cut the first two and the last two qubits are each embedded in a ququart. Therefore, the first and last $R_2$ gate in the quantum circuit 3.12, which act only on the first and last two qubits respectively, become the local ququart gates $R_{01}$ and $R_{23}$, shown in equation (3.10). This can also be seen in the graph 3.13a, where the $R_{01}$ and $R_{23}$ gate are visualized by the red and green edge that are inside the red rectangles, which represent the next-neighbour cut ququart embedding. For better

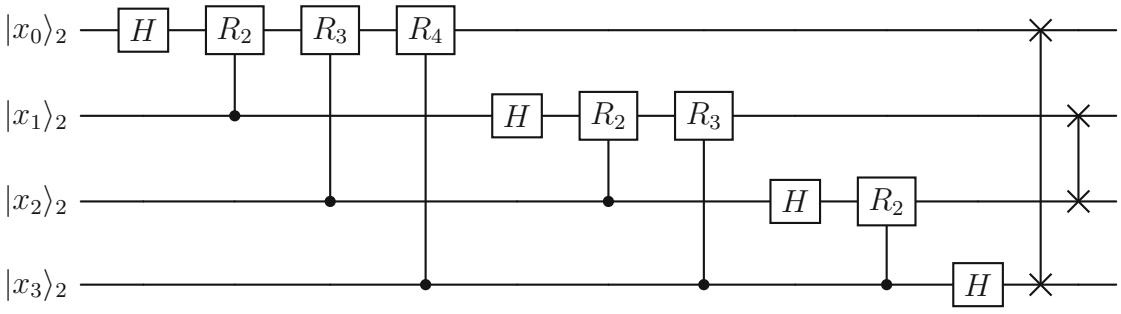
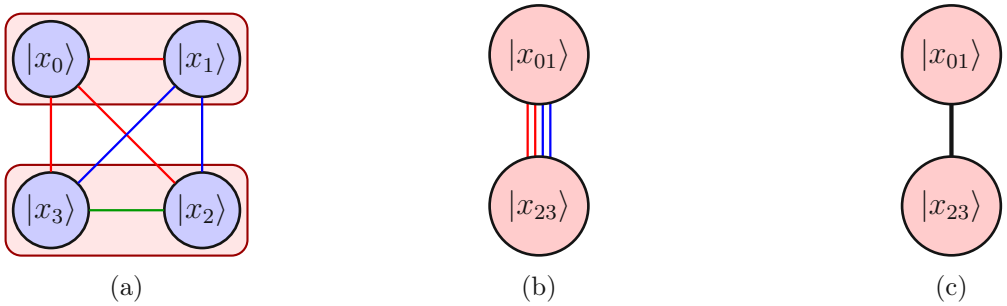



Figure 3.12: Quantum Fourier transform for four qubits




(a) (b) (c)

Figure 3.13: Qudit circuit compression for the $\mathrm{QFT}_4$, where four qubits (blue nodes $|x_i\rangle$) are embedded in two ququarts (red nodes $|x_{ij}\rangle$). In (a), the next-neighbour cut is shown, where two of the six controlled rotation gates become local ququart gates and four remain embedded two-qubit gates, as can be seen in (b). In (c), these four gates are compressed to a single two-ququart gate $R_{01,23}$.

identification of the gates each controlled rotation gate $R_{ij}$ is labeled with two indices, where the first index represents the target qubit and the second one the control qubit. The phase information, which was encoded in the index before, is not lost as the phase index of $\omega_k$ is always $k = j - i + 1$. The other four controlled rotation gates become embedded two-qubit gates, which are constructed of the base matrices $A_i$ and $B_i$, as can be seen in equation (3.11). These four embedded two-qubit gates are represented by the two red and blue edges in the graph 3.13b and the respective quantum circuit for the next-neighbour cut of the $\mathrm{QFT}_4$ is depicted in figure 3.14.

$$R_{01} = R_2 \otimes \mathbb{1}_4 \qquad\qquad R_{23} = \mathbb{1}_4 \otimes R_2 \tag{3.10}$$

$$\begin{aligned} R_{02} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & A_3 & \\ & & & A_3 \end{pmatrix} & R_{03} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & B_4 & \\ & & & B_4 \end{pmatrix} \\ R_{12} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & A_2 & & \\ & & \mathbb{1}_4 & \\ & & & A_2 \end{pmatrix} & R_{13} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & B_3 & & \\ & & \mathbb{1}_4 & \\ & & & B_3 \end{pmatrix} \end{aligned} \tag{3.11}$$

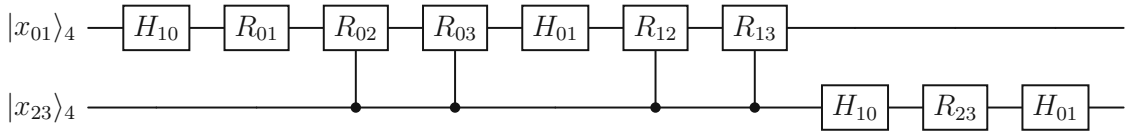



Figure 3.14: Next-neighbour cut for the $\mathrm{QFT}_4$ realized on two ququarts. The controlled rotation gates $R_{01}$ and $R_{23}$ become local ququart gates in the first and second ququart respectively.

The embedded two-qubit gates, which are next to each other in the quantum circuit 3.14, $R_{02}$ and $R_{03}$ and the gates $R_{12}$ and $R_{13}$ can be combined to the two-ququart gates $R_{0,23}$ and $R_{1,23}$, respectively. These gates are built of the $C_{ij}$ base matrices, as can be seen in equation (3.12). The resulting quantum circuit can be seen in figure 3.15.

$$R_{0,23} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & C_{34} & \\ & & & C_{34} \end{pmatrix} \qquad R_{1,23} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & C_{23} & & \\ & & \mathbb{1}_4 & \\ & & & C_{23} \end{pmatrix} \tag{3.12}$$

In order to reduce the remaining two-ququart entangling gates to one, the commutator $[R_{0,23}, H_{01} \otimes \mathbb{1}_4]$ or $[R_{1,23}, H_{01} \otimes \mathbb{1}_4]$ needs to be zero. As the commutator regarding

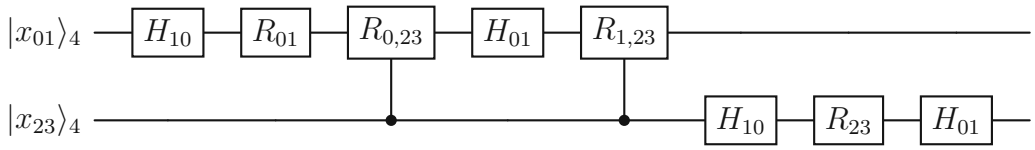



Figure 3.15: Next-neighbour cut for the $\mathrm{QFT}_4$ realized on two ququarts. The embedded two-qubit gates $R_{02}(R_{12})$ and $R_{03}(R_{13})$ can be combined to a two-ququart gate $R_{0,23}(R_{1,23})$.

$R_{0,23}$ is zero, while the other one with $R_{1,23}$ is non-zero, the $R_{0,23}$ gate can only be shifted to the right side. Finally, the two two-ququart gates can be compressed to a single two-ququart gate $R_{01,23}$, shown in equation (3.13). The final quantum circuit is shown in figure 3.16.

$$R_{01,23} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & C_{23} & & \\ & & C_{34} & \\ & & & C_{23}C_{34} \end{pmatrix} \tag{3.13}$$

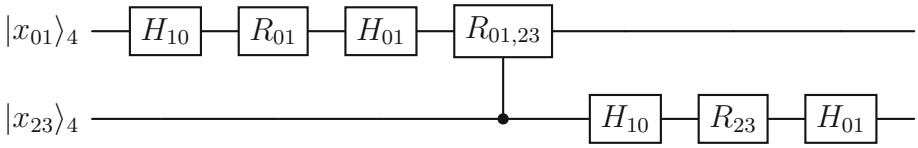



Figure 3.16: Next-neighbour cut for the $\mathrm{QFT}_4$ realized on two ququarts. The commutator $[R_{0,23}, H_{01} \otimes \mathbb{1}_4]$ is zero, therefore the order of these gates can be reversed and the two two-ququart gates can be compressed to a single two-ququart gate $R_{01,23}$.

In conclusion, six two-qubit gates could be reduced to one two-ququart gate and two local ququart gates. This can also be seen in the graphs 3.13b and 3.13c, where four embedded two-qubit gates represented by the red and blue edges are compressed to a single two-ququart gate. As the commutators $[R_{01}, H_{01}]$ and $[R_{23}, H_{10}]$ are non-zero, it is not possible to reduce the circuit even further. Likewise, the local embedded qubit Hadamard gates $H_{10}$ and $H_{01}$ cannot be combined to a local ququart Hadamard gate $H_{11}$ because the commutators $[R_{01}, H_{10}]$ and $[R_{23}, H_{01}]$ are non-zero.

Intuitively, it can be understood that the $R_{0,23}$ gate can be switched with the $H_{01}$ gate while it is impossible with $R_{1,23}$. The $H_{01}$ gate changes the state of the second qubit $|x_1\rangle$ inside the ququart $|x_{01}\rangle$, therefore $R_{0,23}$ can be exchanged with $H_{01}$, since it does not depend on the qubit $|x_1\rangle$. This also applies to aforementioned non-zero commutators to construct local ququart Hadamard gates.

### Inner-outermost cut

In the inner-outermost cut the first and the last qubit and the second and third qubit are each embedded in a ququart, as it is shown in the graph 3.11b. Therefore, the two-qubit gates $R_{03}$ and $R_{12}$ become local ququart gates, shown in equation (3.14). The other four controlled rotation gates become embedded two-qubit gates, which are all constructed by the same base matrices $A_i$ and $B_i$, which can be seen in equation (3.15). The quantum circuit for the inner-outermost cut is shown in figure 3.17.

$$R_{03} = R_4 \otimes \mathbb{1}_4 \qquad\qquad R_{12} = \mathbb{1}_4 \otimes R_2 \tag{3.14}$$

$$\begin{aligned} R_{01} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & A_2 & \\ & & & A_2 \end{pmatrix} & R_{02} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & B_3 & \\ & & & B_3 \end{pmatrix} \\ R_{13} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & A_3 & & \\ & & \mathbb{1}_4 & \\ & & & A_3 \end{pmatrix} & R_{23} &= \begin{pmatrix} \mathbb{1}_4 & & & \\ & B_2 & & \\ & & \mathbb{1}_4 & \\ & & & B_2 \end{pmatrix} \end{aligned} \tag{3.15}$$

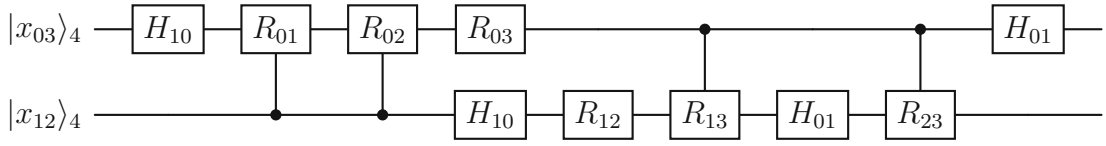



Figure 3.17: Inner-outermost cut for the $\mathrm{QFT}_4$ realized on two ququarts. The controlled rotation gates $R_{03}$ and $R_{12}$ become local gates instead of $R_{01}$ and $R_{23}$, like it is the case for the next-neighbour cut.

The embedded two-qubit gates $R_{01}$ and $R_{02}$ can be directly combined to the two-ququart gate $R_{0,12}$. Since the commutator $[\mathbb{1}_4 \otimes H_{01}, R_{13}]$ is zero, while the other one with $R_{23}$ is non-zero, the $H_{01}$ gate can only be shifted to the left. Consequently, the embedded two-qubit gates $R_{13}$ and $R_{23}$ can be combined to a two-ququart gate $R_{12,3}$. The circuit cannot be further reduced, since the $R_{01}$ gate cannot commute with the $H_{10}$ gate in the second ququart, because both gates act on the first qubit $|x_1\rangle$ embedded in the second ququart $|x_{1,2}\rangle$. Analogously, the $R_{23}$ gate cannot commute with the $H_{01}$ gate in the second ququart, therefore, it is impossible to reduce the number of entangling gates one in this cut, which is verified by all non-zero commutator combinations. Consequently, the next-neighbour cut is superior to the inner-outermost

cut in terms of number of two-ququart gates.

$$R_{0,12} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & \mathbb{1}_4 & & \\ & & C_{23} & \\ & & & C_{23} \end{pmatrix} \qquad R_{12,3} = \begin{pmatrix} \mathbb{1}_4 & & & \\ & C_{23} & & \\ & & \mathbb{1}_4 & \\ & & & C_{23} \end{pmatrix} \tag{3.16}$$

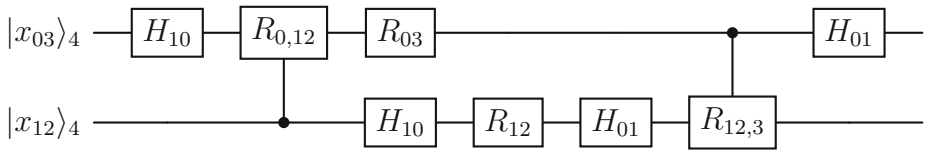



Figure 3.18: Compressed inner-outermost cut for the $\mathrm{QFT}_4$ realized on two ququarts. The embedded two-qubit gates $R_{01}(R_{13})$ and $R_{02}(R_{23})$ can be combined to the two-ququart gate $R_{0,12}(R_{12,3})$.

### Next-next-neighbour cut

The quantum circuit for the next-next-neighbour cut is shown in figure 3.19. There, the controlled rotation gates $R_{02}$ and $R_{13}$ become local ququart gates due to the ququart embedding $|x_{0,2}\rangle$ and $|x_{1,3}\rangle$. The other four two-qubit controlled rotation gates become embedded two-qubit gates and they are equal to the embedded two-qubit gates of the first and second cut, which can be seen in equation (3.17). The $R_{01}$ and $R_{23}$ gate, which become local gates in the first cut, are exactly equal to the respective gates in the second cut. For the remaining two gates $R_{03}$ and $R_{12}$ it is exactly the same just vice versa, since they are equal to the gates of the first cut and are the local gates in the second cut.

$$\begin{aligned} R_{01} &= R_{01\_IO-Cut} & R_{02} &= R_3 \otimes \mathbb{1}_4 & R_{03} &= R_{03\_NN-Cut} \\ R_{12} &= R_{12\_NN-Cut} & R_{13} &= \mathbb{1}_4 \otimes R_3 & R_{23} &= R_{23\_IO-Cut} \end{aligned} \tag{3.17}$$

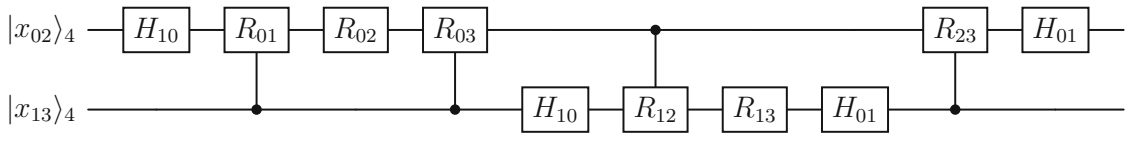



Figure 3.19: Next-next-neighbour cut for the $\mathrm{QFT}_4$ realized on two ququarts. The controlled rotation gates $R_{02}$ and $R_{13}$ become local ququart gates.

The number of entangling gates in the quantum circuit 3.19 cannot be reduced directly, because there are no entangling gates next to each other. In order to compress two entangling gates to one, the entangling gate and the adjacent local gate extended by the identity need to commute, i.e. $[R_{01}, R_{02}] = 0$ and $[R_{12}, (\mathbb{1}_4 \otimes H_{01})R_{13}] = 0$, so that the order of these gates can be reversed. As a result the embedded two-qubit gates $R_{01}$ ($R_{12}$) and $R_{03}$ ($R_{23}$) are aligned next to each other and can be compressed to a two-ququart entangling gate $R_{0,13}$ ($R_{12,23}$). In contrast to all other two-ququart gates so far, the $R_{12,23}$ cannot be illustrated as a gate controlled by a qudit as can be seen in the quantum circuit 3.20, because the second qubit embedded in the first ququart acts as a control and target qubit at the same time.

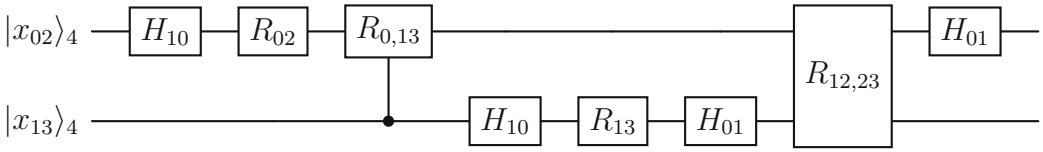



Figure 3.20: Compressed next-next-neighbour cut for the $\mathrm{QFT}_4$ realized on two ququarts. The embedded two-qubit gates $R_{01}(R_{12})$ and $R_{03}(R_{23})$ can be combined to a two-ququart entangling gate $R_{0,13}(R_{12,23})$.

The quantum circuit, shown in figure 3.20, is not further reducible as no combination of gates commutes in a way that the two remaining two-ququart gates can be reduced to a single one. More specifically, the $R_{03}$ gate commutes with $\mathbb{1}_4 \otimes H_{10}$ so it would be possible to combine the gates $R_{03}$ and $R_{12}$ but the remaining embedded two-qubit gates $R_{01}$ and $R_{23}$ would remain, which would result in three remaining entangling gates. This is due to the fact that the $R_{01}$ gate does not commute with the $H_{10}$ gate of the second ququart, nor does the $R_{23}$ gate commute with the $H_{01}$ in the second ququart, because they affect the same qubits.

In conclusion, the quantum circuit for the next-next-neighbour cut, shown in figure 3.20, has the lowest number of two-ququart gates when restricted to the gate set of high-dimensional Hadamard and controlled rotation gates. It is always possible, to express a quantum circuit, which is embedded in only two qudits, with a single two-qudit gate. In the case of the aforementioned quantum circuit, it would be possible to create a unitary gate which includes all local gates which are in between the two two-ququart gates, i.e. $U_{ent} = R_{12,23}(\mathbb{1}_4 \otimes H_{01})R_{13}(\mathbb{1}_4 \otimes H_{10})R_{0,13}$.

This unitary gate $U_{ent}$ is equal to the $\mathrm{DFT}_4$ matrix up to the remaining local gates $H_{10}$, $R_{02}$ and $H_{01}$ in the first ququart, consequently $\mathrm{DFT}_4 = (H_{01} \otimes \mathbb{1}_4)U_{ent}R_{02}(H_{10} \otimes \mathbb{1}_4)$. The entire $\mathrm{QFT}_4$ matrix, which is equivalent to the $\mathrm{DFT}_4$ matrix, as mentioned in section 2.4.2, reduces to a local quhex gate if both ququarts are embedded in one

quhex, which is shown in equation (3.18) for both Hadamard matrix notations.

$$\begin{aligned} \mathrm{QFT}_4 &= H_{0001} R_{23} H_{0010} R_{13} R_{12} H_{0100} R_{03} R_{02} R_{01} H_{1000} \\ &= H_3 R_{23} H_2 R_{13} R_{12} H_1 R_{03} R_{02} R_{01} H_0 \end{aligned} \tag{3.18}$$

In order to generalize this approach to N-qubits, it is necessary to understand the structure of the high-dimensional controlled rotation gates $R_{ij}$. Therefore, the analysis of the $\mathrm{QFT}_6$ and the $\mathrm{QFT}_8$ is necessary, but as the number of cuts vastly increases by increasing the number of qubits, only the next-neighbour cut, which performed best in the case of the $\mathrm{QFT}_4$, is taken into consideration.

### 3.3.3 6-qubit QFT

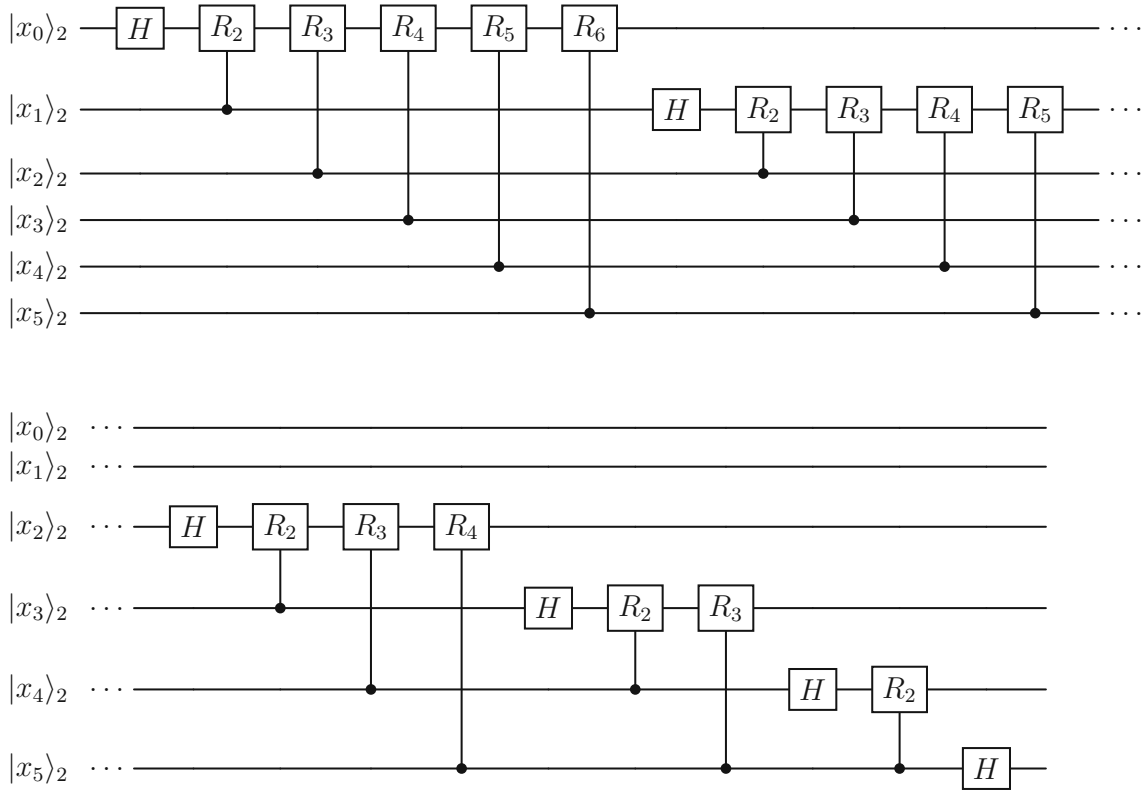
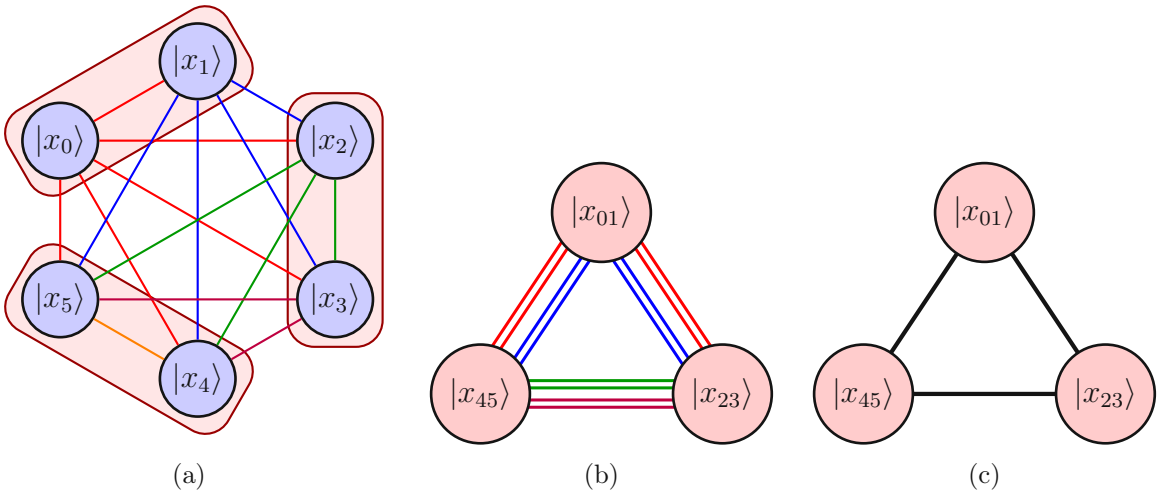



Figure 3.21: Quantum Fourier transform for six qubits

The quantum circuit for the $\mathrm{QFT}_6$, shown in figure 3.21, consists of local qubit Hadamard matrices $H$ and controlled rotation matrices $R_i$. Since $2^6 = 64$ distinct states are possible, the $\mathrm{QFT}_6$ could be embedded in a 64-dimensional quantum system, but due to the

fact that addressing 64 distinct quantum states in a single qudit poses a serious experimental challenge, this approach is impracticable at the moment. As one ququart (quoct) embeds four (eight) different states, three ququarts $4^3 = 64$ (two quocts $8^2 = 64$) are needed to embed the 64 different states. It would also be possible to embed the $\mathrm{QFT}_6$ in one ququart and one quhex $4 \times 16 = 64$ but in the following subsections, the $\mathrm{QFT}_N$ is only discussed in pure embeddings, which means that the embedding only consists of one type of qudit, because these are the most relevant use cases.

### $\mathrm{QFT}_6$ embedded in three ququarts




(a) (b) (c)

Figure 3.22: Qudit circuit compression for the $\mathrm{QFT}_6$. Six qubits (blue nodes $|x_i\rangle$) are embedded in three ququarts (red nodes $|x_{ij}\rangle$). In (a), the next-neighbour cut is shown, where three of the 15 controlled rotation gates become local ququart gates and twelve remain embedded two-qubit gates as can be seen in (b). The red, blue, green and purple edges represent $R_{0j}$, $R_{1j}$, $R_{2j}$ and $R_{3j}$ gates, respectively. In (c), four embedded two-qubit gates each are compressed to a single two-ququart gate.

The quantum circuit for the $\mathrm{QFT}_6$ embedded in three ququarts in the next-neighbour cut is shown in figure 3.23, where the controlled rotation gates $R_{01}$, $R_{23}$ and $R_{45}$ become local ququart gates due to the cut specific embedding $\{0, 1\}$, $\{2, 3\}$ and $\{4, 5\}$. This can also be seen in the graph 3.22a, where two qubits $|x_i\rangle$ (blue nodes) are each embedded in a ququart $|x_{ij}\rangle$ (red rectangles). All gates (edges), which are inside a red rectangle which represents the next-neighbour cut ququart embedding, become local ququart gates, which are the red, green and orange edge $R_{01}$, $R_{23}$ and $R_{45}$, respectively. In 3.22a, it is observable that the QFT represents a fully connected graph, where each edge represents a controlled rotation gate with the target qubit $|x_i\rangle$ and the control qubit $|x_j\rangle$, where $i < j$ holds for any gate. The remaining twelve embedded two-qubit gates can

be seen in figure 3.22b. In the following section, it is proven that these twelve gates can be compressed to only three two-ququart gates, which represent a fully connected graph again, as can be seen in figure 3.22c.

All high-dimensional controlled rotation gates that have a shape of $64 \times 64$ follow a specific structure, as can be seen in the following equations. In order to shorten the notation in these equations, the notation $X^{\otimes y} = \bigotimes_{i=0}^{y-1} X$ is adapted to $X^{\otimes y} = \mathbb{1}_y \otimes X = \text{block\_diag}_y(X)$. The matrices below are shown in greater resolution in appendix B.

$$R_{01} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_{16} & \\ & \omega^2\mathbb{1}_{16} \end{pmatrix} \end{pmatrix} \quad R_{02} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_8 & \\ & \omega^3\mathbb{1}_8 \end{pmatrix}^{\otimes 2} \end{pmatrix} \quad R_{03} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_4 & \\ & \omega^4\mathbb{1}_4 \end{pmatrix}^{\otimes 4} \end{pmatrix} \quad R_{04} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_2 & \\ & \omega^5\mathbb{1}_2 \end{pmatrix}^{\otimes 8} \end{pmatrix} \quad R_{05} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} 1 & \\ & \omega^6 \end{pmatrix}^{\otimes 16} \end{pmatrix}$$

$$R_{12} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} \mathbb{1}_8 & \\ & \omega^2\mathbb{1}_8 \end{pmatrix} \end{pmatrix}^{\otimes 2} \quad R_{13} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} \mathbb{1}_4 & \\ & \omega^3\mathbb{1}_4 \end{pmatrix}^{\otimes 2} \end{pmatrix}^{\otimes 2} \quad R_{14} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} \mathbb{1}_2 & \\ & \omega^4\mathbb{1}_2 \end{pmatrix}^{\otimes 4} \end{pmatrix}^{\otimes 2} \quad R_{15} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} 1 & \\ & \omega^5 \end{pmatrix}^{\otimes 8} \end{pmatrix}^{\otimes 2}$$

$$R_{23} = \begin{pmatrix} \mathbb{1}_8 & & \\ & \begin{pmatrix} \mathbb{1}_4 & \\ & \omega^2\mathbb{1}_4 \end{pmatrix} \end{pmatrix}^{\otimes 4} \quad R_{24} = \begin{pmatrix} \mathbb{1}_8 & & \\ & \begin{pmatrix} \mathbb{1}_2 & \\ & \omega^3\mathbb{1}_2 \end{pmatrix}^{\otimes 2} \end{pmatrix}^{\otimes 4} \quad R_{25} = \begin{pmatrix} \mathbb{1}_8 & & \\ & \begin{pmatrix} 1 & \\ & \omega^4 \end{pmatrix}^{\otimes 4} \end{pmatrix}^{\otimes 4}$$

$$R_{34} = \begin{pmatrix} \mathbb{1}_4 & & \\ & \begin{pmatrix} \mathbb{1}_2 & \\ & \omega^2\mathbb{1}_2 \end{pmatrix} \end{pmatrix}^{\otimes 8} \quad R_{35} = \begin{pmatrix} \mathbb{1}_4 & & \\ & \begin{pmatrix} 1 & \\ & \omega^3 \end{pmatrix}^{\otimes 2} \end{pmatrix}^{\otimes 8}$$

$$R_{45} = \begin{pmatrix} \mathbb{1}_2 & & \\ & \begin{pmatrix} 1 & \\ & \omega^2 \end{pmatrix} \end{pmatrix}^{\otimes 16}$$

The controlled rotation gates $R_{0j}$ shown above are block diagonal matrices composed of a 32-dimensional identity matrix and a base matrix $A_{abc} = diag(\mathbb{1}_a, \omega^b\mathbb{1}_a)^{\otimes c}$, where the indices $a$, $b$ and $c$ depend on the indices $i$ and $j$ of $R_{ij}$ and $N$. The controlled rotation gates $R_{1j}$ are block diagonal matrices composed of two blocks of $B_{abcd}$, which is again a block diagonal matrix of a 16-dimensional identity matrix and the base matrix $A_{abc}$, i.e. $B_{abcd} = diag(\mathbb{1}_d, A_{abc})$. This can be continued for the matrices $R_{2j}$, $R_{3j}$ and $R_{45}$, which implicates that the dimensionality of the identity matrix $d$ of $B_{abcd}$ decreases with increasing index $i$, while the number of blocks of $B_{abcd}$ increases with $i$. The index $a$ of the base matrix $A_{abc}$ decreases as $j$ increases, while the index $c$ increases with $j$. The index $b$ which specifies the phase $\omega$ depends on both $i$ and $j$, more specifically $b$ increases with index $j$ but decreases with $i$. The exact relationship between the indices $a$, $b$, $c$, $d$ and $i$, $j$ and $N$ is given in equation (3.19). Therefore, $R_{ij}(N)$ can be expressed as a combination of identity matrices and $\omega$, which depend on the indices $i$ and $j$ and the total number of qubits $N$, which is 6 for the $\text{QFT}_6$.

$$R_{ij}(N) = \mathbb{1}_{2^i} \otimes \begin{pmatrix} \mathbb{1}_x & & \\ & \mathbb{1}_{2^k} \otimes \begin{pmatrix} \mathbb{1}_y & \\ & \omega_k\mathbb{1}_y \end{pmatrix} \end{pmatrix} \text{ with } \begin{cases} x = 2^{N-i-1} \\ y = 2^{N-j-1} \\ k = j - i + 1 \end{cases} \tag{3.19}$$

Equation (3.19) does not only hold for the $\text{QFT}_6$ but for any number of input qubits $N$. Furthermore, the formula does not make any restrictions regarding the dimensionality of the qudits involved, which means that it can be used for the construction of any high-dimensional controlled rotation gate $R_{ij}$ in an arbitrary qudit embedding in

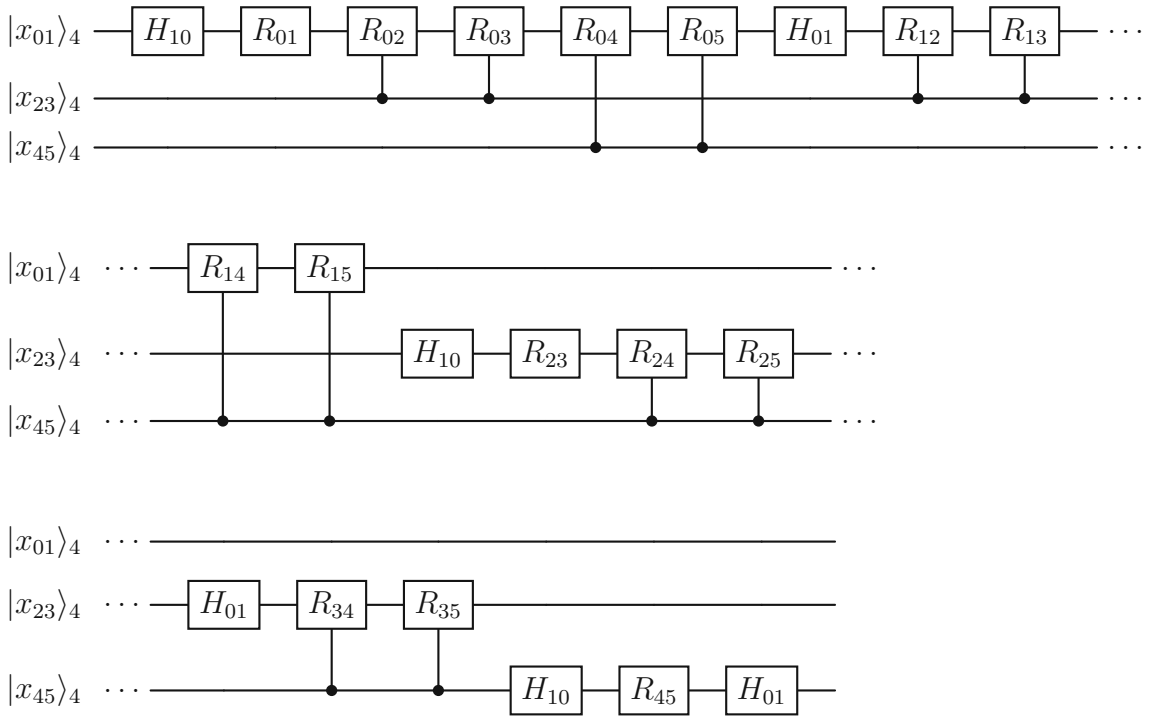



Figure 3.23: Next-neighbour cut for the $\mathrm{QFT}_6$ realized on three ququarts. The controlled rotation gates $R_{01}$, $R_{23}$ and $R_{45}$ become local ququart gates due to the next-neighbour-cut embedding.

the next-neighbour cut. This formula is used throughout this thesis, especially in the upcoming sections 3.3.4 and 3.3.5.

Due to the symmetry of the QFT in the next-neighbour cut, two embedded two-qubit gates, which affect the same ququarts always align next to each other, hence can be compressed to a two-ququart gate, which yields quantum circuit 3.24.

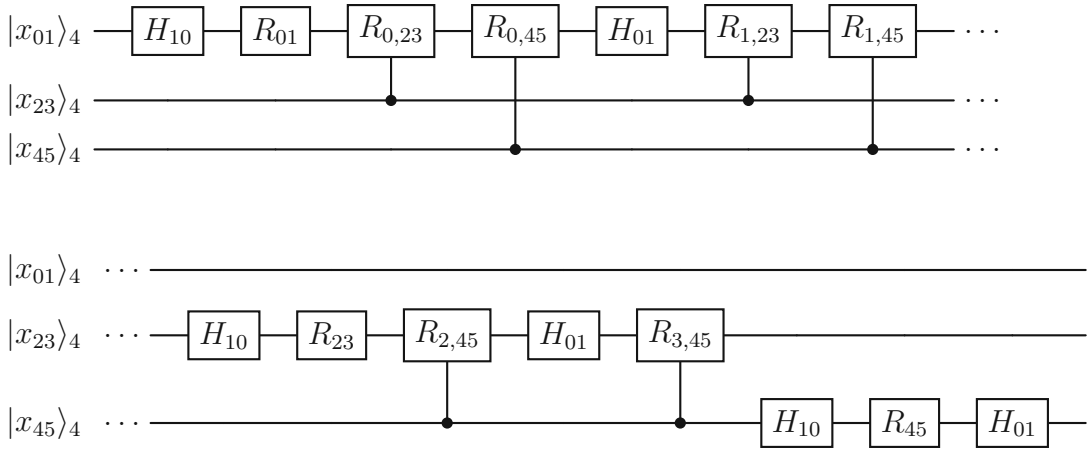



Figure 3.24: Compressed next-neighbour cut for the $\mathrm{QFT}_6$ realized on three ququarts. Adjacent embedded two-qubit gates which affect the same ququarts can be compressed to a single two-ququart gate, e.g. $R_{0,23} = R_{03}R_{02}$.

In order to further reduce the number of two-ququart gates, gates acting on the same ququart must be compressed. However, since no two-ququart gates are adjacent anymore, it must be verified that they commute with the intermediate gates. $R_{0,23}$ commutes with $R_{0,45}$ because they are both diagonal matrices and $[R_{0,23}, H_{01} \otimes \mathbb{1}_{16}]$ is also zero because $H_{01}$ acts only on the second qubit $|x_1\rangle$ inside the first ququart $|x_{01}\rangle$ and $R_{0,23}$ acts only on the qubits $\{0, 2, 3\}$ inside the first two ququarts. Therefore, $R_{0,23}$ can be combined with $R_{1,23}$ to the two-ququart gate $R_{01,23} = R_{1,23}R_{0,23}$. However, it would have not been possible to shift the $R_{1,23}$ gate to the left of $H_{01}$ because both gates act on the second qubit $|x_1\rangle$. For the two-ququart gate pair $R_{0,45}$ and $R_{1,45}$, it is not possible to move the $R_{1,45}$ gate to the left side of $H_{01}$, but $R_{0,45}$ can be interchanged with $H_{01}$ for the same reason as before, which yields the gate $R_{01,45} = R_{1,45}R_{0,45}$. The same applies to the gates $R_{2,45}$ and $R_{3,45}$, because of the commutators $[R_{2,45}, \mathbb{1}_4 \otimes H_{01} \otimes \mathbb{1}_4] = 0$ and $[R_{3,45}, \mathbb{1}_4 \otimes H_{01} \otimes \mathbb{1}_4] \neq 0$, which yields the gate $R_{23,45} = R_{3,45}R_{2,45}$. All three compressed two-ququart gates $R_{ij,kl}$ encompass four two-qubit gates each, which is also shown in figure 3.22. The compressed quantum circuit, where all these commutator relations were used, is shown in figure 3.25.

Gates which affect only the first two ququarts $R_{0,23}$, $R_{1,23}$ and $R_{01,23}$ are equal to the corresponding gates of the $\mathrm{QFT}_4$ up to the extension of the identity, i.e. $R_{01,23\_QFT_6} =$

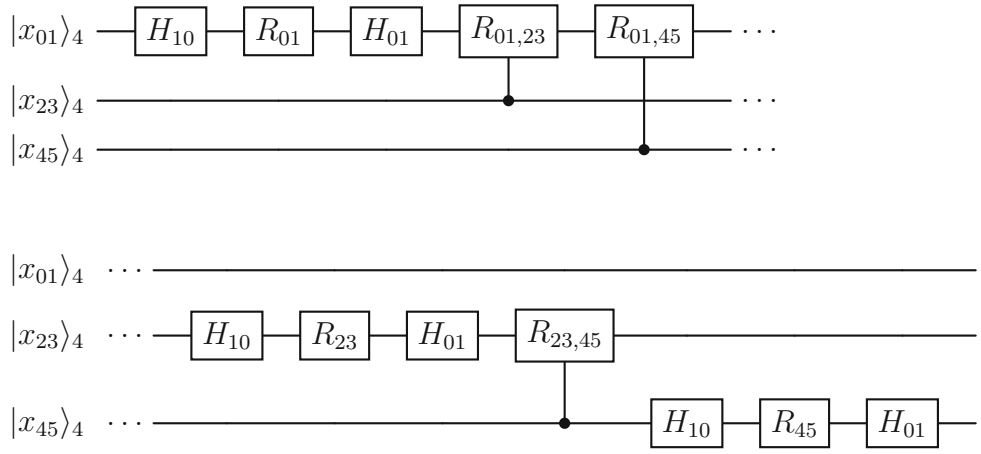



Figure 3.25: Next-neighbour cut for the $\mathrm{QFT}_6$ realized on three ququarts. Two-ququart gates $R_{0,23}$, $R_{0,45}$ and $R_{2,45}$ can be combined with $R_{1,23}$, $R_{1,45}$ and $R_{3,45}$ respectively, which yields the two-ququart gates $R_{01,23}$, $R_{01,45}$ and $R_{23,45}$ respectively, because the commutators with intermediate gates are zero.

$R_{01,23\_QFT_4} \otimes \mathbb{1}_4$.

In summary, there are 15 two-qubit gates in the qubit embedding, which reduce to twelve embedded two-qubit gates and three local ququart gates by the next-neighbour cut of the ququart embedding. These twelve embedded two-qubits gates are then compressed to six two-ququart gates, which can then be reduced to only three two-ququart gates by applying the commutator rules.

### $\mathrm{QFT}_6$ embedded in two quocts

The graphical representation of the next-neighbour cut of the quoct embedding can be seen in figure 3.26, where the blue and green nodes represent qubits $|x_i\rangle$ and quocts $|x_{ijk}\rangle$, respectively and the edges represent controlled rotation gates. For each quoct, three two-qubit gates are transformed into a local quoct gate. These are represented for the first quoct by the two red edges $R_{01}$ and $R_{02}$ and the blue edge $R_{12}$, and for the second quoct by the two purple edges $R_{34}$ and $R_{35}$ and the orange edge $R_{45}$, as illustrated in figure 3.26a. Therefore, 15 two-qubit gates become six local quoct gates and nine are transformed to embedded two-qubit gates in a quoct embedding, contrary to the embedded two-qubit gates in a ququart embedding. The remaining nine gates between the two quocts can be compressed to a single two-quoct gate, which is shown in the figures 3.26b and 3.26c, which is proven in the following.

The quantum circuit, that represents the graph 3.26b, is shown in figure 3.27. The adjacent embedded two-qubit gates $R_{i3}$, $R_{i4}$, and $R_{i5}$ can be compressed to two-quoct gates $R_{i,345}$. Therefore, the number of entangling gates is reduced from nine to three,

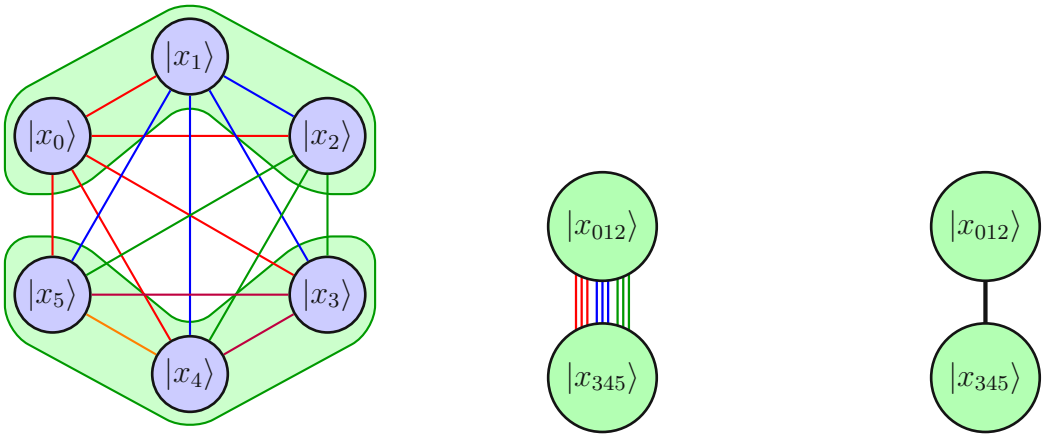



Figure 3.26: Qudit circuit compression for the $\mathrm{QFT}_6$ in the quoct embedding. Six qubits (blue nodes $|x_i\rangle$) are embedded in two quocts (green nodes $|x_{ijk}\rangle$). In (a), the next-neighbour cut is shown, where six of the 15 controlled rotation gates become local quoct gates and nine remain embedded two-qubit gates, as can be seen in (b). The red, blue and green edges represent $R_{0j}$, $R_{1j}$ and $R_{2j}$ gates, respectively. In (c), nine embedded two-qubit gates are compressed to a single two-quoct gate.

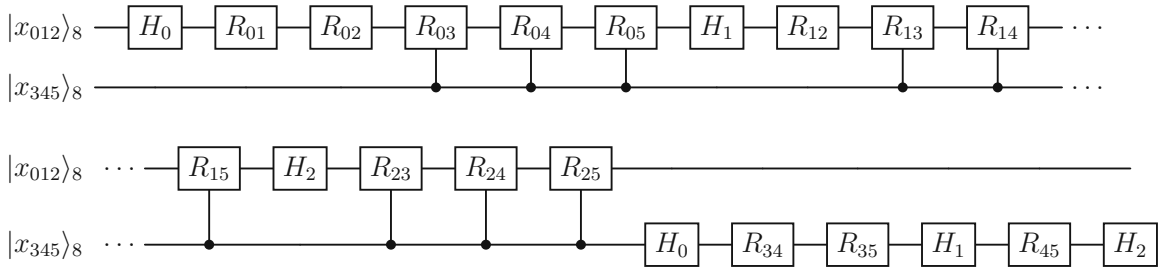



Figure 3.27: Next-neighbour cut for the $\mathrm{QFT}_6$ embedded in two quocts. Controlled rotation gates that only include the indices $\{0, 1, 2\}$ and $\{3, 4, 5\}$ become local quoct gates in the first and second quoct, respectively. Gates that include indices of both sets are non-local embedded two-qubit gates.

which is shown in figure 3.28.

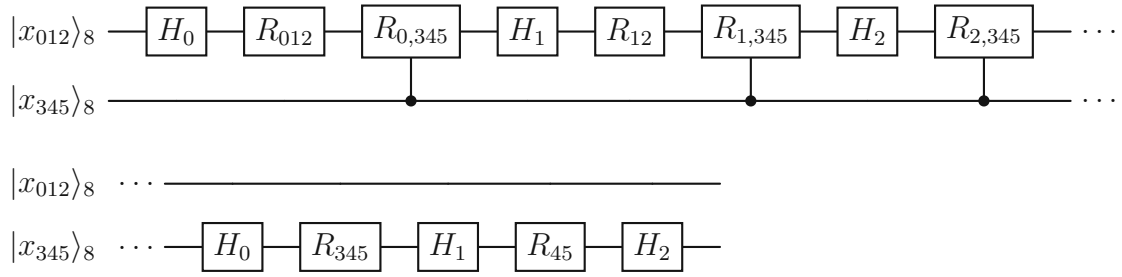



Figure 3.28: Compressed next-neighbour cut for the $\mathrm{QFT}_6$ embedded in two quocts. Adjacent emebedded two-qubit gates $R_{i3}$, $R_{i4}$ and $R_{i5}$ can be compressed a two-quoct gate $R_{i,345}$.

To reduce the number of entangling gates to one, it is required that $R_{0,345}$ commutes with both $H_1$ and $H_2$ and that $R_{1,345}$ commutes with $H_2$. As $R_{0,345}$ does not act on the qubits 1 and 2 inside the first quoct, it commutes with both Hadamard gates $H_1$ and $H_2$. The commutator $[R_{1,345}, H_2 \otimes \mathbb{1}_8]$ is also zero because $R_{1,345}$ does not act on the second qubit. Therefore, all two-quoct gates can be compressed to a single two-quoct gate $R_{012,345} = R_{2,345}R_{1,345}R_{0,345}$, which yields the quantum circuit in figure 3.29.

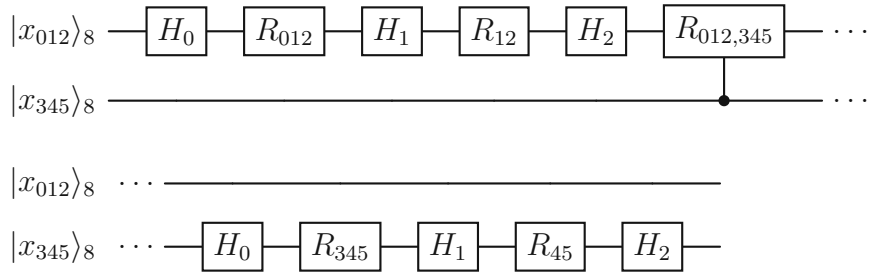
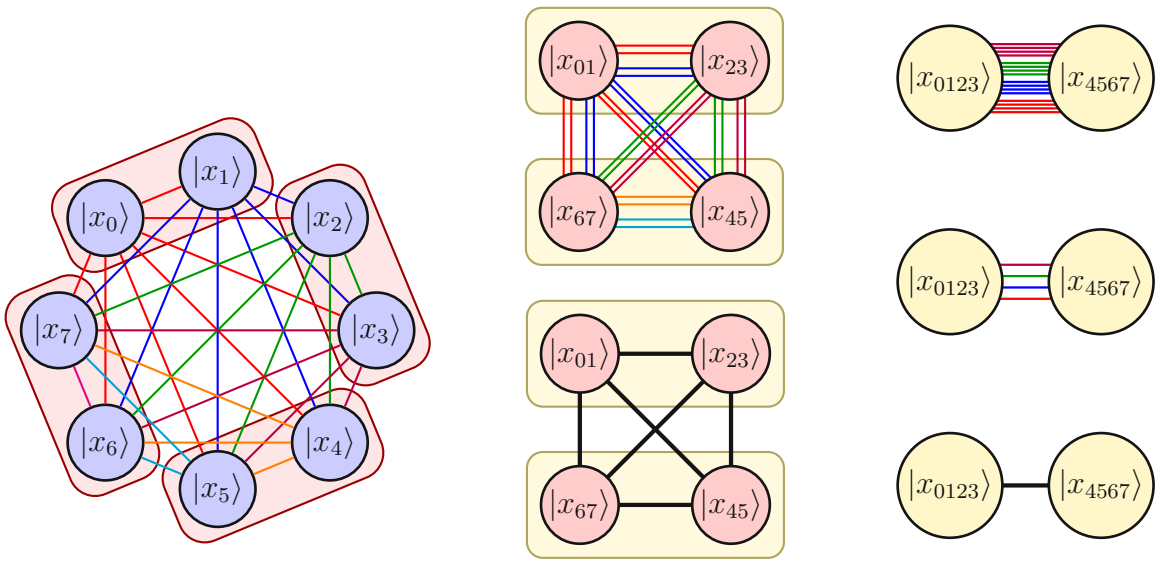



Figure 3.29: Compressed next-neighbour cut for the $\mathrm{QFT}_6$ embedded in two quocts. The two-qucot gates $R_{0,345}$ and $R_{1,345}$ can be shifted to the right, since they commute with all local gates in between, which yields a single compressed two-quoct gate $R_{012,345}$.

As already seen, in the ququart embedding of the $\mathrm{QFT}_4$ and $\mathrm{QFT}_6$ it is always possible to shift the controlled rotation gates to the right, but it is always impossible to shift the gates to the left. This is also the case for the $\mathrm{QFT}_6$ in the quoct embedding. In the next section, it is examined if this trend continues for the $\mathrm{QFT}_8$, but also if it is always possible to reduce the number of entangling gates to one for each qudit pair.

### 3.3.4 8-qubit QFT

The quantum circuit for the $\mathrm{QFT}_8$ in the qubit embedding is not explicitly shown, but as the general quantum circuit for the $\mathrm{QFT}_N$ is given in figure 2.3, the derivation




(a) (b) (c)

Figure 3.30: Qudit circuit compression for the $\mathrm{QFT}_8$ in the next-neighbour cut, where in (a) eight qubits (blue nodes $|x_i\rangle$) are embedded in four ququarts (red rectangles). 28 controlled rotation gates become four local ququart gates and 24 remain embedded two-qubit gates. In (b), four ququarts (red nodes $|x_{ij}\rangle$) are embedded in two quhexes (yellow rectangles). There, four embedded two-qubit gates become one two-ququart gate, which yields a fully connected graph for four ququarts connected by six two-ququart gates. In (c), two quhexes can execute the $\mathrm{QFT}_8$ by either 16 embedded two-qubit gates, four embedded two-ququart gates or a single two-quhex gate. It is important to note that the four black edges in the bottom graph in (b) are embedded two-ququart gates in two quhexes, while the middle graph in (c) shows four two-quhex gates.

of the corresponding circuit is trivial. The $QFT_8$ entails $2^8 = 256$ distinct states, which means it could be executed with a single unitary matrix $DFT_8$ which possesses $256 \times 256 = 65536$ entries, but due to aforementioned reasons this is impractical. Therefore, it can be embedded in four ququarts $4^4 = 256$ or in two quhexes $16^2 = 256$. It is impossible to embed the $QFT_8$ only in quocts as one quoct embedds three qubits, but eight is not divisible by three without remainder. As $N$ increases the number of possible embeddings explodes as it is possible to combine qudits of all dimensions. Therefore, it would be possible to use one quhex and two ququarts or two quocts and one ququart or even one quhex, one quoct and one qubit.

In the following only those embeddings, which utilize only one type of qudit, which are only the ququart and quhex embeddings for the $QFT_8$, are investigated in the next-neighbour cut. These embeddings are shown in figure 3.30, where the blue, red and yellow nodes represent the qubits $|x_i\rangle$, the ququarts $|x_{ij}\rangle$ and the quhexes $|x_{ijkl}\rangle$, respectively. It can be observed that the QFT consistently yields a fully connected graph across all embeddings in the next-neighbour cut.

### $QFT_8$ embedded in four ququarts

The next-neighbour cut of the $QFT_8$ embedded in four ququarts is shown in the quantum circuit 3.31. Due to next-neighbour cut embedding every controlled rotation gate $R_{ij}$ becomes a local ququart gate, if it fulfills the conditions $j - i = 1$ and $i \mod 2 = 0$, which yields the gates $R_{01}$, $R_{23}$, $R_{45}$ and $R_{67}$. This quantum circuit represents the top graph in figure 3.30b, where all embedded two-qubit gates are shown. Due to the next-neighbour cut ququart embedding, there are always two embedded two-qubit gates adjacent, therefore they can be compressed to a two-ququart gate, e.g. $R_{0,23} = R_{03}R_{02}$. This halves the amount of entangling gates and by shifting the controlled rotation gates to the right of $H_{01}$, which is always possible, all gates that affect the same ququarts can be combined, which halves the amount of entangling gates again. The resulting quantum circuit is shown in figure 3.32, which is equivalent to the bottom graph in figure 3.30b.

In conclusion, 28 two-qubit gates become 4 local ququart gates and the remaining 24 embedded two-qubit gates are reduced by the factor of four to six two-ququart gates, which are graphically represented as the six edges in the fully connected graph, shown in the bottom figure 3.30b.

In the following section, the $QFT_8$ is analyzed in the context of the next-neighbour cut for four ququarts embedded within two quhexes, in order to assess whether the same principles of qudit circuit compression — from four qubits to two ququarts — also apply to higher-dimensional systems.

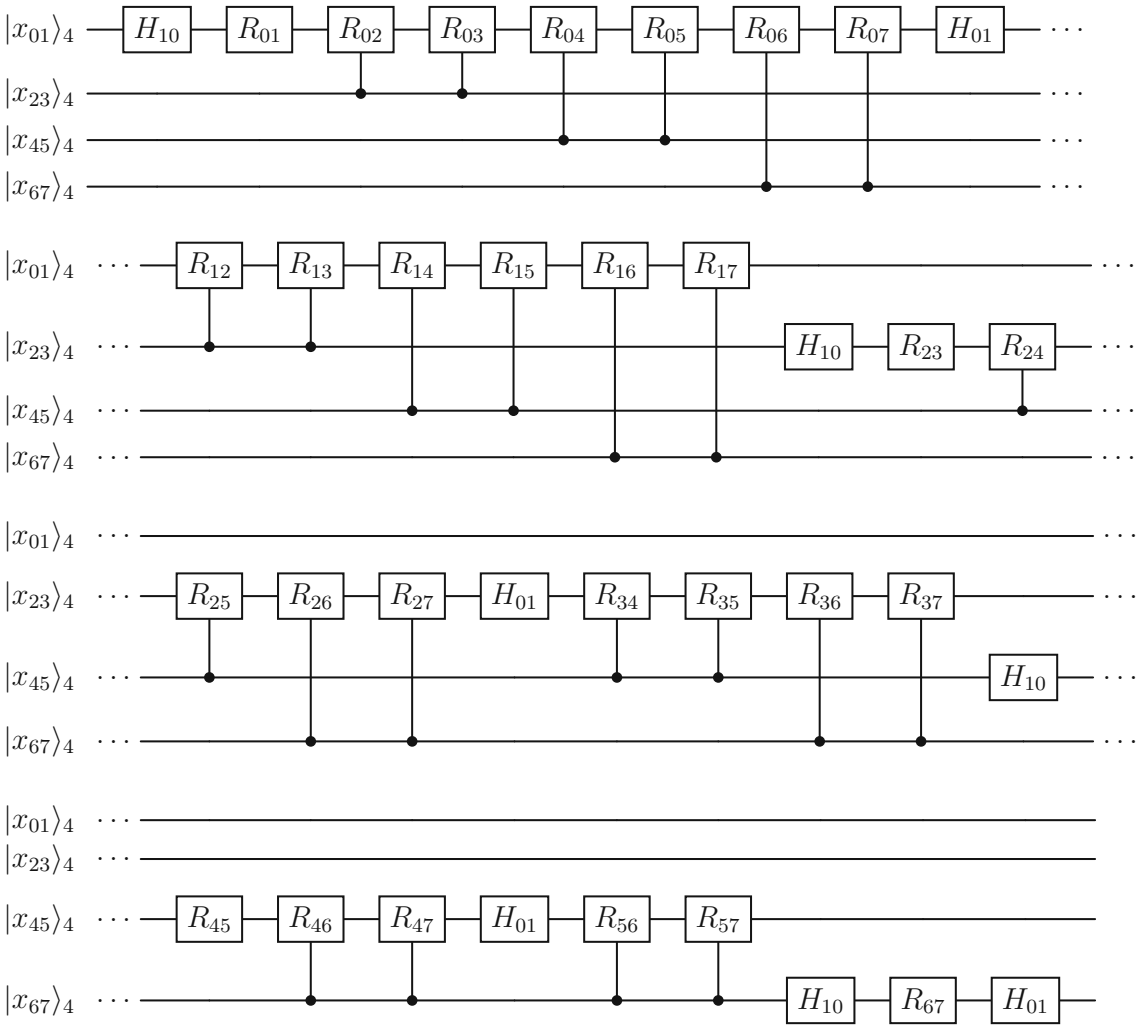



Figure 3.31: Next-neighbour cut of the $\mathrm{QFT}_8$ realized on four ququarts. 28 two-qubit gates become four local ququart gates $R_{01}$, $R_{23}$, $R_{45}$ and $R_{67}$ due to the next-neighbour-cut embedding and all other $R_{ij}$ gates represent embedded two-qubit gates.

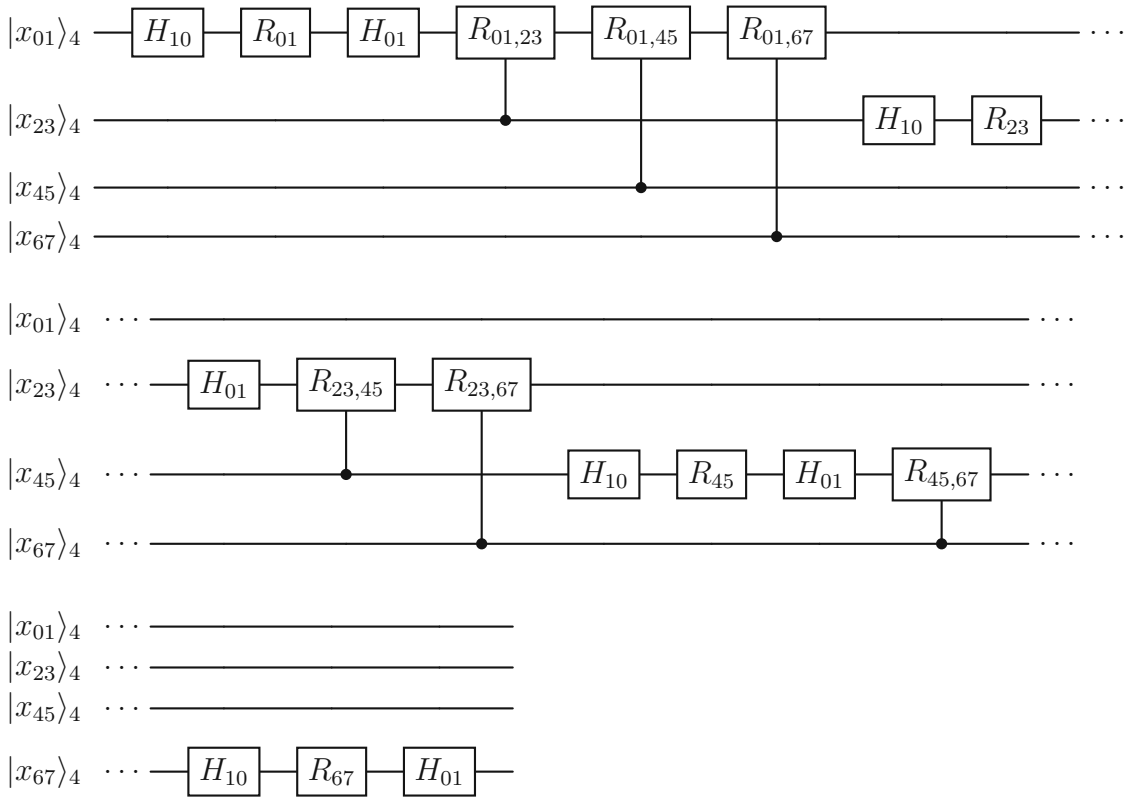



Figure 3.32: Compressed next-neighbour cut for the $\mathrm{QFT}_8$ realized on four ququarts. Adjacent controlled rotation gates are compressed and can be shifted to the right of $H_{01}$, therefore, they can be combined with the two-ququart gate that affects the same ququarts. This reduces the number of entangling gates by a factor of four.

### $\mathrm{QFT}_8$ embedded in two quhexes

The next-neighbour cut of the $\mathrm{QFT}_8$ for four ququarts is shown in figure 3.30b, where the red nodes represent the ququarts $|x_{ij}\rangle$, the yellow rectangles the next-neighbour cut quhex embedding and the edges represent embedded two-qubit gates and embedded two-ququart gates for the top and bottom graph, respectively. The next-neighbour cut quhex embedding yields the quhexes $|x_{0123}\rangle$ and $|x_{4567}\rangle$. The controlled rotation gates manifest as either 16 embedded two-qubit gates, four embedded two-ququart gates or one two-quhex gate, which can be seen in figure 3.30.

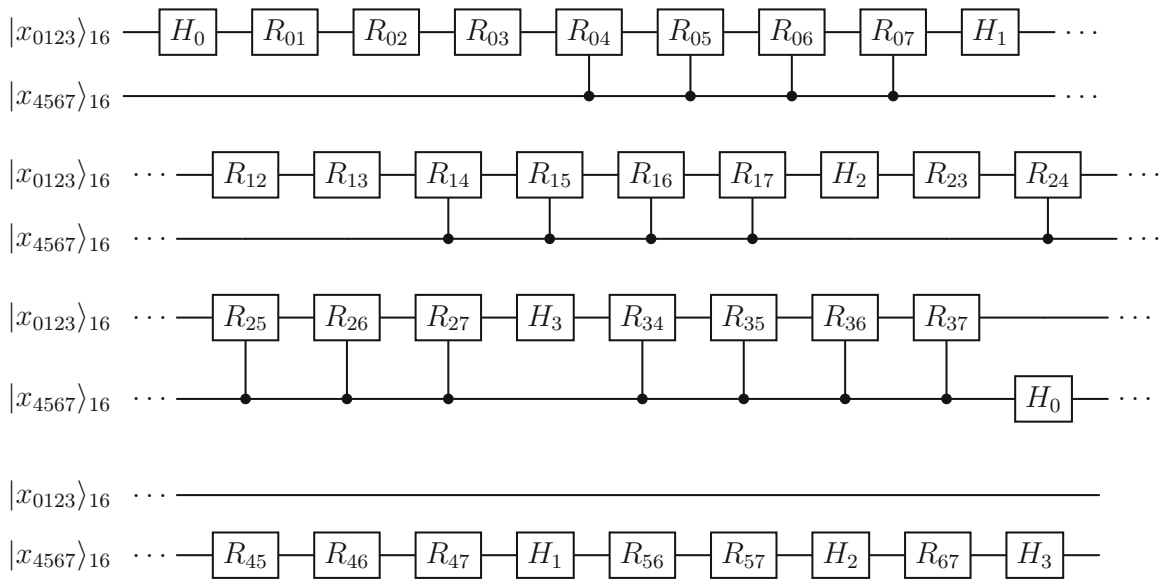



Figure 3.33: Next-neighbour cut for the $\mathrm{QFT}_8$ embedded in two quhexes. Controlled rotation gates that only include the indices $\{0,1,2,3\}$ and $\{4,5,6,7\}$ become local quhex gates in the first and second quhex, respectively. Gates that include indices of both sets manifest as embedded two-qubit gates.

The quantum circuit that corresponds to the top graph of figure 3.30c is shown in figure 3.33. There, the four edges for each color represent the 16 embedded two-qubit gates $R_{i4}$, $R_{i5}$, $R_{i6}$ and $R_{i7}$, where $i \in \{0,1,2,3\}$ belongs to the red, blue, green and purple edges, respectively. The ququarts $|x_{01}\rangle$ and $|x_{23}\rangle$ in the top graph of figure 3.30b, are encompassed by the yellow rectangle (quhex embedding), which implicates that the $R_{02}$, $R_{03}$ and $R_{12}$, $R_{13}$ gates become local quhex gates, which are shown as the red and blue edges, respectively. Same applies to the local quhex gates $R_{46}$, $R_{47}$ and $R_{56}$, $R_{57}$ which represent the orange and cyan edges between $|x_{45}\rangle$ and $|x_{67}\rangle$. The local quhex gates $R_{01}$ ($R_{45}$) and $R_{23}$ ($R_{67}$) for the first (second) quhex are not shown figure 3.30b, because they are already local in the respective ququart and can only be seen as the

four edges inside the red rectangles (ququart embedding) of figure 3.30a, hence they can also be seen as embedded local ququart gates.

The number of entangling gates of this quantum circuit can be reduced by multiplying adjacent embedded two-qubit gates $R_{i,4567} = \prod_{j=4}^{7} R_{ij}$, which yields the quantum circuit 3.34. This quantum circuit is equivalent to the middle graph of figure 3.30c where the red, blue, green and purple edge represent the gates $R_{0,4567}$, $R_{1,4567}$, $R_{2,4567}$ and $R_{3,4567}$, respectively. It is important to note that these four gates represent two-quhex gates, whereas the four black edges that connect the yellow rectangles (quhex embedding) in the bottom graph of figure 3.30b are embedded two-ququart gates. The difference between those gates is, that the two-quhex gate acts on both ququarts inside one of the participating quhexes, while the embedded two-ququart gate acts only on one ququart inside one quhex. The edges of the embedded two-ququart gates are black, because they are a mixture of two colors, for instance the embedded two-ququart gate between the ququarts $|x_{01}\rangle$ and $|x_{45}\rangle$ is the combination of two red and blue edges, which represents the gate $R_{01,45} = R_{15}R_{14}R_{05}R_{04}$. In order to achieve a corresponding quantum circuit, which only uses embedded two-ququart gates it would be necessary to reorder all gates, which feature the same ququarts, next to each other by shifting them to the right of the Hadamard gates $H_i$. Although this circuit also possesses four entangling gates it is inherently different than the quantum circuit 3.34, because it uses two-quhex gates instead of embedded two-ququart gates.

Adjacent local controlled rotation gates can be combined to a single local quhex gate, which consists of three and two local gates for the first and second qubit inside the first quhex, respectively, i.e. $R_{0123} = R_{03}R_{02}R_{01}$ and $R_{123} = R_{13}R_{12}$. The local quhex gates of the second quhex are constructed in the same way.

As already mentioned in previous sections, it is always possible to shift the controlled rotation gates to the right of $H_i$ in the next-neighbour cut, because they do not affect the same qubits inside the qudit. Therefore, all four remaining two-quhex gates can be combined to a single two-quhex gate $R_{0123,4567} = \prod_{j=0}^{3} R_{j,4567}$, which yields the quantum circuit 3.35. It is important to note, that the controlled rotation matrices defined above with the product symbol $\prod$, are in the wrong order, but as all controlled rotation gates are diagonal matrices, they commute and therefore the order is irrelevant. In order to ensure the correct order of non-commuting gates in the next section, a decreasing product $\coprod$ is defined.

In summary, 28 two-qubit gates are reduced to 16 embedded two-qubit gates and 12 local gates by the quhex embedding. The remaining 16 gates are reduced by the factor of four by combining adjacent gates to two-quhex gates. The remaining four two-quhex gates can be compressed to a single two-quhex gate by rearranging the order of the gates by using the commutator rules.

In the previous sections, it is observable that the qudit circuit compression of the QFT in the next-neighbour cut follows specific rules. The number of entangling gates

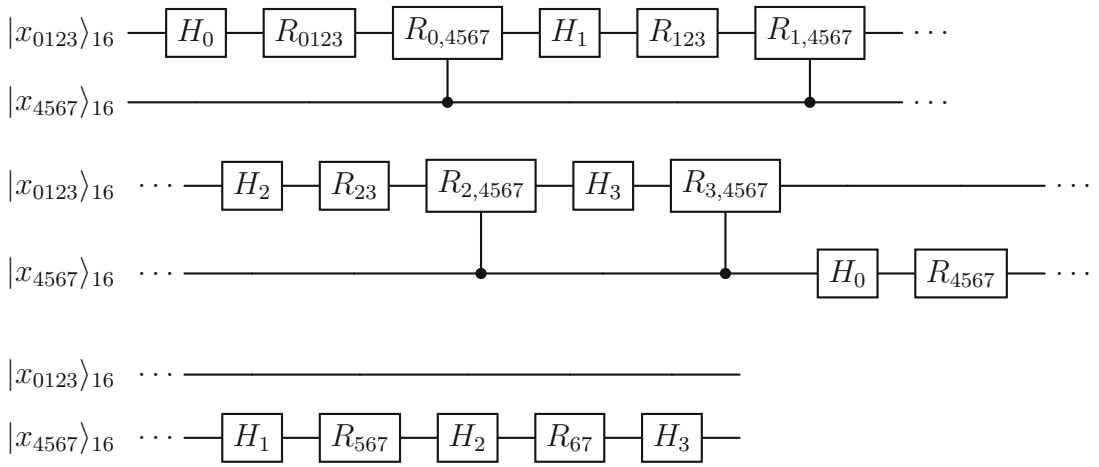



Figure 3.34: Compressed next-neighbour cut for the $\mathrm{QFT}_8$ embedded in two quhexes. Four adjacent embedded two-qubit gates are each compressed to two-quhex gates $R_{i,4567} = \prod_{j=4}^{7} R_{ij}$, with $i \in \{0, 1, 2, 3\}$. Local controlled rotation gates can be compressed to a local quhex gate, e.g. $R_{0123} = R_{03}R_{02}R_{01}$.

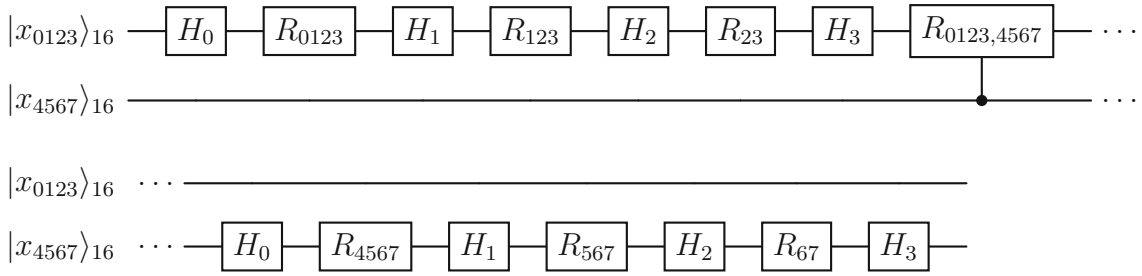



Figure 3.35: Compressed next-neighbour cut for the $\mathrm{QFT}_8$ embedded in two quhexes. All two-quhex gates $R_{i,4567}$ can be shifted to the right, because they commute with intermediate Hadamard gates $H_i$, which yields the two-quhex gate $R_{0123,4567} = \prod_{j=0}^{3} R_{j,4567}$.

depends on the number of qubits involved $N$ and the dimension of the qudit embedding. In the upcoming section 3.3.5, the general structure of qudit circuit compression of the $\mathrm{QFT}_N$ is discussed. Based on these results, a formula for the number of entangling gates is derived in section 3.3.6.

### 3.3.5 N-qubit QFT

The symmetry of the QFT in the next-neighbour cut yields a fully connected graph for all dimensions. This means that the QFT embedded in any type of qudits can be always represented as a fully connected graph until $d = 2^N$, because at this point the QFT can be executed on a single qudit, which would represent a single node and not a fully connected graph. The fully connected graphs in figure 3.30, which do not use more than one edge per qudit pair, represent the $\mathrm{QFT}_{16}$ when replacing qubits by ququarts, ququarts by quhexes and quhexes by a 256-dimensional qudit. With this kind of recursion it is possible to create all possible graphs for any number of qubits $N$ embedded in any qudit dimension $d$.

In order to formally prove that the $\mathrm{QFT}_N$ executed on qubits is equivalent to the $\mathrm{QFT}_N$ executed on qudits, it is necessary to formulate an equation that relates these two approaches. The $\mathrm{DFT}_N$ matrix, shown in equation 2.19, describes the $\mathrm{QFT}_N$ for $N$ bits, therefore the $\mathrm{DFT}_N$ matrix can be equated with the multiplication of all gates of the quantum circuit. This is already shown for the $\mathrm{QFT}_4$ in equation (3.18), where the entire $\mathrm{QFT}_4$ is performed with a single local quhex gate. In this equation is important to notice that the first gate of the quantum circuit is the last gate in the equation. In order to respect the correct order of execution of the gates, it is necessary to define a decreasing product $\coprod$, where the symbol of the coproduct $\coprod$ is used, which is shown in equation (3.20).

$$\coprod_{i=N}^{M} x_i = x_N x_{N-1} x_{N-2} \dots x_M \text{ with } M < N \tag{3.20}$$

The structure of the QFT requires two decreasing products, since there are $N-1$ blocks, where each one consists of one Hadamard gate $H_i$ and multiple $R_{ij}$ gates. Since the controlled rotation matrices are diagonal — and therefore commute — it is not necessary to use a decreasing product for the $R_{ij}$ gates. Nevertheless, a decreasing product is employed to illustrate the underlying formalism. The $H_i$ gates were defined up to now as the local Hadamard gate, which act only on the respective qudit but in order to multiply it with the controlled rotation gates $R_{ij}$, which span over the entire Hilbert space, it is necessary to extend the $H_i$ gate by the identity matrix. As an example the $H_0$ gate on the second quhex in the quantum circuit in figure 3.35 of the $\mathrm{QFT}_8$ embedded in two quhexes would be $H_{0,new} = \mathbb{1}_{16} \otimes H_0 = H_4$, which is exactly the continuation of the definition, given in equation (2.9). Therefore, it is possible to

use this definition of $H_i$ with $i \in [0, N]$ in equation (3.21), which defines the $\mathrm{QFT}_N$ in terms of high dimensional Hadamard and controlled rotation matrices.

$$\mathrm{QFT}_N = \coprod_{i=N-1}^{0} \left( \coprod_{j=N-1}^{i+1} R_{ij} \right) H_i \tag{3.21}$$

To express the preceding equation in standard notation using an increasing product, the order of the product can be reversed through reindexing $i \to k = N - (i - M) = N + M - i$, which is shown in equation (3.22).

$$\coprod_{i=N}^{M} x_i = \coprod_{k=M}^{N} x_{N+M-k} \tag{3.22}$$

Using equation (3.22) two times on equation (3.21) yields equation (3.23), where the indices $i$ and $j$ are replaced by $k$ and $l$.

$$\mathrm{QFT}_N = \prod_{k=0}^{N-1} \left( \prod_{l=N-k}^{N-1} R_{(N-k-1)(2N-l-k-1)} \right) H_{N-k-1} \tag{3.23}$$

In the previous sections, SWAP gates were neglected due to aforementioned reasons, but in order to match the $\mathrm{DFT}_N$ matrix, SWAP gates need to be applied to the $\mathrm{QFT}_N$. One way to define a high dimensional multi-qudit SWAP gate that acts on the entire Hilbert space, which encapsulates all SWAP gates of the $\mathrm{QFT}_N$, is a recursive formula, shown in equation (3.24). There, each entry of the SWAP(N) gate is 0 except for one entry in each row $i \in [0, 2^N]$, where $j(i)$ fulfills the recursive condition $j(i) = j(i-a) + 2^N/(2a)$, where $a$ uses the floor function $\lfloor x \rfloor$.

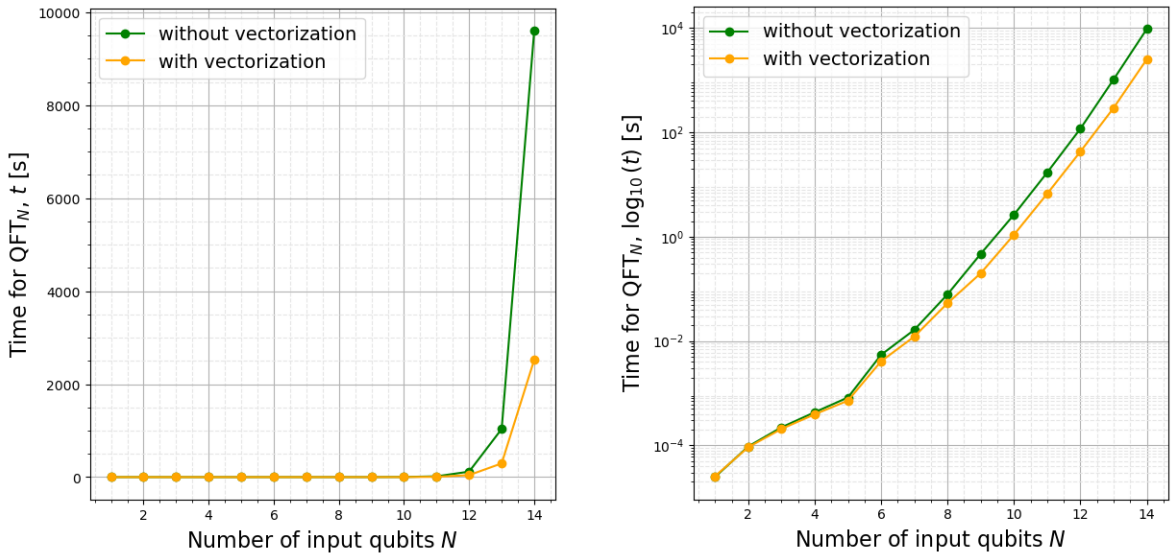
$$\mathrm{SWAP}_{ij}(N) = 1 \text{ with } j(i) = j(i-a) + \frac{2^N}{2a}, j(0) = 0, a = 2^{\lfloor \log_2 i \rfloor}, i \in \left[0, 2^N\right] \tag{3.24}$$

The $\mathrm{QFT}_N$ matrix, given in equation (3.23), is equal to the $\mathrm{DFT}_N$ matrix up to the SWAP(N) gate, therefore, the SWAP(N) gate is multiplied from the left onto the $\mathrm{QFT}_N$ matrix, which yields equation (3.25).

$$\mathrm{SWAP(N)} \cdot \prod_{k=0}^{N-1} \left( \prod_{l=N-k}^{N-1} R_{(N-k-1)(2N-l-k-1)} \right) H_{N-k-1} = \mathrm{DFT}_N = F_N \tag{3.25}$$

Although the formula for the $R_{ij}$ matrices was constructed only for the $\mathrm{QFT}_{2n}$, above equation was proven for all $n$ numerically up to $N = 14$. The matrices for $N = 14$ are already of size $2^{14} \times 2^{14} = 16384 \times 16384$, consequently the computation on a classical computer is too time-consuming to prove it for higher $N$. In figure 3.36, it

is illustrated that the time required for the calculation of the $\mathrm{QFT}_N$ matrix, increases exponentially with the system size $N$, where a linear and logarithmic scaling is shown in figures (a) and (b), respectively. The green line represents the calculation through standard matrix multiplication of formula (3.25), while the orange line exploits the fact that the $R_{ij}$ matrices are diagonal matrices and can therefore efficiently multiplied by an element-wise product of the corresponding vectors.




(a) (b)

Figure 3.36: The time required for the $\mathrm{QFT}_N$ presented with a linear scaling (a) and a logarithmic scaling (b), illustrating the exponential increase in computational cost with growing system size. The green line calculates the QFT through standard matrix multiplication, while the orange line exploits the diagonal structure of the $R_{ij}$ matrices.

### 3.3.6 Number of two-qudit gates

The number of qudits $q$ of dimension $d$, which are necessary to perform a QFT for $N$ input qubits, is given by $q = N/\log_2(d)$. Due to the symmetry of the QFT in the next-neighbour cut, the number of two-qudit gates $g_E(N, d) \equiv g_d(N)$ can be expressed as a function of the input qubit count $N$ and the qudit dimension $d$. In order to deduce the structure of the formula for $g_E$, it is useful to recap all the results from the previous sections, which are shown in table 3.1. There, the number of (embedded) two-qubit gates $g_d$, the number of two-qudit gates compressed through adjacent gates $g_{d-A}$ and the number of two-qudit gates $g_{d-C}$, achieved through gate reordering, are shown. The number of gates for the $\mathrm{QFT}_4$, $\mathrm{QFT}_6$ and $\mathrm{QFT}_8$ can be obtained by counting the

edges in the corresponding figures. The number of entangling gates for the $\mathrm{QFT}_{12}$ and $\mathrm{QFT}_{16}$ can be determined analogously using the graph-based approach described earlier, however, the derivation is not shown explicitly for $N = 12$ and $N = 16$.

| N | $g_2$ | $g_4$ | $g_{4-A}$ | $g_{4-C}$ | $g_8$ | $g_{8-A}$ | $g_{8-C}$ | $g_{16}$ | $g_{16-A}$ | $g_{16-C}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 6 | 4 | 2 | 1 | – | – | – | 0 | 0 | 0 |
| 6 | 15 | 12 | 6 | 3 | 9 | 3 | 1 | – | – | – |
| 8 | 28 | 24 | 12 | 6 | – | – | – | 16 | 4 | 1 |
| 12 | 66 | 60 | 30 | 15 | 54 | 18 | 6 | 48 | 12 | 3 |
| 16 | 120 | 112 | 56 | 28 | – | – | – | 96 | 24 | 6 |

Table 3.1: Number of entangling gates $g_d$ of the QFT in the next-neighbor cut as a function of the input qubit count $N$ and the qudit dimension $d$. $g_d$ represents the (embedded) two-qubit gates, $g_{d-A}$ represents the number of gates achieved through compressing adjacent gates and $g_{d-C}$ describes the number of gates achieved through gate reordering by using the commutator rules.

The number of entangling gates for the $\mathrm{QFT}_N$ embedded in qubits is given by equation (3.26), as already mentioned in section 2.4.2. There, the number of two-qubit gates $g_d(N)$ does not depend on the qudit dimension $d = 2$, since it depends only on $N$. This equation yields the number of two-qubit gates $g_2(N)$, shown in the first column in table 3.1.

$$g_2(N) = \sum_{i=1}^{N-1} i = \frac{N(N-1)}{2} \tag{3.26}$$

The number of entangling gates in the ququart embedding $g_4$ is reduced by one for each ququart, as one ququart transforms one two-qubit gate into a local ququart gate. As already mentioned in section 3.3.2, the number of entangling gates is halved by combining adjacent gates $g_{4-A}$ and is halved again by compressing two-ququart gates via gate reordering, which yields $g_{4-C}$. Equation (3.27) captures this behavior, as the number of qudits $q$ is subtracted from the original amount of two-qubit gates $g_2$, which is then halved twice by the factor of $2^2$.

$$\begin{aligned} g_4(N, d = 4) &= g_2(N) - q(N, d = 4) = \frac{N(N-1)}{2} - \frac{N}{\log_2(4)} = \frac{N(N-2)}{2} \\ g_{4-C}(N, d = 4) &= \frac{N(N-2)}{2 \cdot 2^2} \end{aligned} \tag{3.27}$$

The quoct embedding reduces three two-qubit gates to local quoct gates, therefore, it is necessary to subtract $3q$ gates from the original amount $g_2$. As can be seen in figure 3.26, the number of entangling gates is reduced by the factor $3^2$, which yields equation

(3.28).

$$
\begin{aligned}
g_8(N, d=8) &= g_2(N) - 3q(N, d=8) = \frac{N(N-1)}{2} - \frac{3N}{\log_2(8)} = \frac{N(N-3)}{2} \\
g_{8-C}(N, d=8) &= \frac{N(N-3)}{2 \cdot 3^2}
\end{aligned}
\tag{3.28}
$$

In the quhex embedding six two-qubit gates are reduced to local quhex gates, therefore it is necessary to subtract $6q$, subsequently the number of entangling gates is reduced by the factor four, two times, which gives equation (3.29).

$$
\begin{aligned}
g_{16}(N, d=16) &= g_2(N) - 6q(N, d=16) = \frac{N(N-1)}{2} - \frac{6N}{\log_2(16)} = \frac{N(N-4)}{2} \\
g_{16-C}(N, d=16) &= \frac{N(N-4)}{2 \cdot 4^2}
\end{aligned}
\tag{3.29}
$$

It is already possible to deduce the structure of the formula for the number of two-qudit gates $g_E(N, d)$ by replacing the values 2, 3 and 4 in $g_{4-C}$, $g_{8-C}$ and $g_{16-C}$ respectively, with $\log_2(d)$. It is also possible to derive the formula for $g_E(N, d)$, because the prefactors of $q$ are given by the column $g_2$. This makes sense as the qudits always embed a fully connected graph, which consists of one, three, six, ten, ... edges for a ququart, quoct, quhex, qudit (d=32), ... , respectively. This prefactors which correspond to the edges of a fully connected graph are also known as triangular numbers [37]. The formula for the number of edges in a fully connected graph with $N$ nodes is given by $g_2(N)$ in equation (3.26). The expression $g_2(\log_2(d))$ yields the number of two-qubit gates, which are reduced to local qudit gates. Subsequently, the number of embedded two-qubit gates is reduced by the factor of $\log_2(d)^2$, because there are always $\log_2(d)$ adjacent gates and $\log_2(d)$ gates, which can be combined through gate reordering. The resulting expression for $g_E(N, d)$ is given in equation (3.30), where it is possible through arithmetic simplification to derive the already anticipated formula for $g_E(N, d)$.

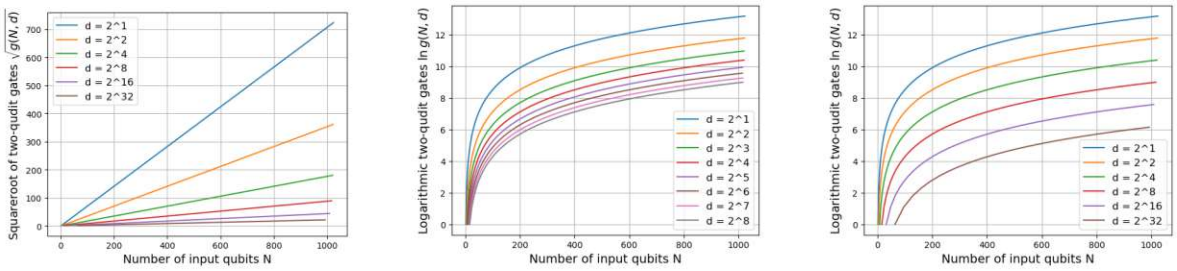
$$
\begin{aligned}
g_E(N, d) &= \left[g_2(N) - g_2(\log_2(d)) \cdot q\right] / \log(d)^2 \\
&= \left(\frac{N(N-1)}{2} - \frac{\log_2(d)(\log_2(d)-1)}{2} q\right) / \log_2(d)^2 \\
&= \frac{N(N-1) - \frac{N \log_2(d)(\log_2(d)-1)}{\log_2(d)}}{2 \log_2(d)^2} \\
&= \frac{N(N-1) - N(\log_2(d)-1)}{2 \log_2(d)^2} \\
&= \frac{N(N - \log_2(d))}{2 \log_2(d)^2}
\end{aligned}
\tag{3.30}
$$

The formula for $g_E(N,d)$ always yields the values of a fully connected graph, because the QFT in the next-neighbour cut always represents a fully connected graph in any qudit embedding. If $N$ and $d$ are not compatible with one another, which means that the $\mathrm{QFT}_N$ cannot be embedded in this type of qudit because of dimensionality issues, $g_E(N,d)$ yields a non-integer value. For $d = 2^N$, $g_E(N,d)$ is always zero because the entire $\mathrm{QFT}_N$ can be embedded in one local qudit gate. This behaviour can be observed in table 3.2, where the first column represents the triangular numbers, but they can also be found in all other columns separated by $\log_2(d) - 1$ non-integer values, which are represented by a minus sign. The embedding is only possible if $N/\log_2(d)$ is an integer value, which is equivalent to the number of qudits $q$.

| N | $g_2$ | $g_4$ | $g_8$ | $g_{16}$ | $g_{32}$ | $g_{64}$ | $g_{128}$ | $g_{256}$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | - | - | - | - | - | - | - |
| 2 | 1 | 0 | - | - | - | - | - | - |
| 3 | 3 | - | 0 | - | - | - | - | - |
| 4 | 6 | 1 | - | 0 | - | - | - | - |
| 5 | 10 | - | - | - | 0 | - | - | - |
| 6 | 15 | 3 | 1 | - | - | 0 | - | - |
| 7 | 21 | - | - | - | - | - | 0 | - |
| 8 | 28 | 6 | - | 1 | - | - | - | 0 |
| 9 | 36 | - | 3 | - | - | - | - | - |
| 10 | 45 | 10 | - | - | 1 | - | - | - |
| 11 | 55 | - | - | - | - | - | - | - |
| 12 | 66 | 15 | 6 | 3 | - | 1 | - | - |
| 16 | 120 | 28 | - | 6 | - | - | - | 1 |
| 24 | 276 | 66 | 28 | 15 | - | 6 | - | 3 |
| 32 | 496 | 120 | - | 28 | - | - | - | 6 |
| 64 | 2016 | 496 | - | 120 | - | - | - | 28 |

Table 3.2: Listing of the number of entangling gates $g_E(N,d)$ for different $N$ and $d$ values. In all rows and columns the triangular numbers can be observed, where the triangular numbers are separated by $\log_2(d) - 1$ non-integer values, represented by a minus sign, which makes sense because the qudit embedding is only possible for $\log_2(d)n$.

The graphical representation of $g_E(N,d)$ calculated for all valid pairs is shown in figure 3.37. In (a), the quadratic scaling of $g_E(N,d)$ can be observed as a linear line induced by the square root $\sqrt{g_E(N,d)}$ scaling used in this plot. In (b), a logarithmic scaling of $g_E(N,d)$ is shown, which illustrates that the reduction of entangling gates becomes less when increasing the exponent $x$ of $d = 2^x$ linearly, while the reduction factor stays constant when increasing the exponent exponentially $d = 2^{2^x}$, shown in figure (c).




(a) (b) (c)

Figure 3.37: Graphical representation of $g_E(N, d)$ calculated for all valid pairs $N$ and $d$. In (a), a square root scaling of $g_E(N, d)$ is shown, which illustrates the quadratic behaviour. In (b) and (c), a logarithmic scaling illustrates the behaviour of the reduction of entangling gates for different qudit dimensions $d$.

In conclusion, the next-neighbour cut is the best cut for the QFT, because it is always possible to reduce the number of entangling gates between two qudits to a single two-qudit gate. The number of embedded two-qubit gates is given by $g_E(N, d)$ without the divisor $\log_2(d)^2$ and the number of two-qudit gates is given by $g_E(N, d)$.

# Chapter 4

# Summary and outlook

Qudit based quantum computing provides several advantages over qubit based quantum computing, where in this thesis the reduction of entangling gates achieved by a qudit embedding was examined. The qudit circuit compression of the 4-qubit Simon's algorithm was investigated, where it was possible to reduce the number of entangling gates always to a single two-ququart gate irrespective of the chosen cut.

The QFT was examined in detail, where it was possible to deduce, that the next-neighbour cut is the best cut for the QFT, as it reduces all gates that affect the same qudits to a single two-qudit gate. Furthermore, it was possible to ascertain that the QFT in the next-neighbour cut can be represented by a fully connected graph in any qudit embedding considered. Moreover, it was possible to derive a formula for the number of entangling gates as a function of the number of input qubits $N$ and the qudit dimension $d$. The formula for the QFT decomposed as high dimensional controlled rotation and Hadamard gates was proven numerically to be equal to the DFT matrix for all $n$ up to $N = 14$.

As the QFT is only a subroutine in most quantum algorithms, it would be interesting if quantum algorithms, like Shor's algorithm, also perform best in the next-neighbour cut. It would also be possible to analyze the high dimensional controlled rotation gates in terms of entanglement power. Furthermore, it would be interesting if these high dimensional controlled rotation gates can be easily implemented experimentally. Otherwise, one could try to find a decomposition of the QFT in terms of native two-qudit gates, which are available in experiments at the moment.

# Bibliography


[1] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[2] John Preskill. "Quantum computing in the NISQ era and beyond". In: *Quantum* 2 (2018), p. 79.

[3] Martin Ringbauer et al. "Certification and quantification of multilevel quantum coherence". In: *Physical Review X* 8.4 (2018), p. 041007.

[4] Tristan Kraft et al. "Characterizing genuine multilevel entanglement". In: *Physical review letters* 120.6 (2018), p. 060502.

[5] Fern HE Watson et al. "Qudit color codes and gauge color codes in all spatial dimensions". In: *Physical Review A* 92.2 (2015), p. 022312.

[6] Earl T Campbell. "Enhanced fault-tolerant quantum computing in d-level systems". In: *Physical review letters* 113.23 (2014), p. 230501.

[7] Max Planck. "Zur Theorie des Gesetzes der Energieverteilung im Normalspektrum". In: *Berlin* (1900), pp. 237–245.

[8] Max Born and Pascual Jordan. "Zur Quantenmechanik". In: *Zeitschrift für Physik* 34.1 (1925), pp. 858–888.

[9] Werner Heisenberg. "Über quantentheoretische Umdeutung kinematischer und mechanischer Beziehungen." In: *Zeitschrift für Physik* (1925), pp. 879–893.

[10] V Dorobantu. "The postulates of quantum mechanics". In: (2006). arxiv preprint physics/0602145.

[11] Paul Adrien Maurice Dirac. "A new notation for quantum mechanics". In: *Mathematical proceedings of the Cambridge philosophical society*. Vol. 35. 3. Cambridge University Press. 1939, pp. 416–418.

[12] Richard Jozsa. "Entanglement and Quantum Computation". In: (1997). arXiv: quant-ph/9707034.

[13] Eli Biham et al. "Quantum computing without entanglement". In: *Theoretical Computer Science* 320.1 (2004), pp. 15–33.

[14] Dan Kenigsberg, Tal Mor, and Gil Ratsaby. "Quantum advantage without entanglement." In: *Quantum Inf. Comput.* 6.7 (2006), pp. 606–615.

[15] JE Savage. "Models of computation. Exploring the power of computing Addison-Wesley". In: *Reading, MA* (1998).

[16] John E Hopcroft, Rajeev Motwani, and Jeffrey D Ullman. "Introduction to automata theory, languages, and computation". In: *Acm Sigact News* 32.1 (2001), pp. 60–65.

[17] Michael Sipser. "Introduction to the Theory of Computation". In: *ACM Sigact News* 27.1 (1996), pp. 27–29.

[18] Donald E Knuth. *The Art of Computer Programming: Seminumerical Algorithms*. Vol. 2. Addison-Wesley Professional, 1969. Chap. 4.1, p. 186.

[19] Wilfried Buchholz. "Fingers or fists?(the choice of decimal or binary representation)". In: *Communications of the ACM* 2.12 (1959), pp. 3–11.

[20] Peter W Shor. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer". In: *SIAM review* 41.2 (1999), pp. 303–332.

[21] Stephane Beauregard. "Circuit for Shor's algorithm using 2n+ 3 qubits". In: *arXiv preprint quant-ph/0205095* (2002).

[22] David P. DiVincenzo. "Topics in Quantum Computers". In: (1996). arXiv: cond-mat/9612126.

[23] Wojciech Hubert Zurek. "Decoherence and the transition from quantum to classical—revisited". In: *Quantum decoherence: poincaré seminar 2005*. Springer. 2005, pp. 1–31.

[24] A Yu Kitaev. "Fault-tolerant quantum computation by anyons". In: *Annals of physics* 303.1 (2003), pp. 2–30.

[25] Laszlo Gyongyosi and Sandor Imre. "A survey on quantum computing technology". In: *Computer Science Review* 31 (2019), pp. 51–71.

[26] Richard P Feynman. "Quantum mechanical computers." In: *Found. Phys.* 16.6 (1986), pp. 507–532.

[27] Roger Penrose et al. "Applications of negative dimensional tensors". In: *Combinatorial mathematics and its applications* 1 (1971), pp. 221–244.

[28] Adriano Barenco et al. "Elementary gates for quantum computation". In: *Physical review A* 52.5 (1995), p. 3457.

[29] Alastair Kay. "Tutorial on the quantikz package". In: (2018). arXiv preprint arXiv:1809.03842.

[30] Lov K Grover. "A fast quantum mechanical algorithm for database search". In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996, pp. 212–219.

[31] David Deutsch and Richard Jozsa. "Rapid solution of problems by quantum computation". In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907 (1992), pp. 553–558.

[32] Ethan Bernstein and Umesh Vazirani. "Quantum complexity theory". In: *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*. 1993, pp. 11–20.

[33] Daniel R Simon. "On the power of quantum computation". In: *SIAM journal on computing* 26.5 (1997), pp. 1474–1483.

[34] Gilbert Strang. *Introduction to linear algebra*. SIAM, 2022.

[35] Alan V Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999.

[36] Xiaoqin Gao et al. "On the role of entanglement in qudit-based circuit compression". In: *Quantum* 7 (2023), p. 1141.

[37] Ronald L Graham. *Concrete mathematics: a foundation for computer science*. Pearson Education India, 1994.

# List of Figures

# List of Tables

# Appendix A

# QFT product representation

The QFT maps the state $|x\rangle$ to a superposition of $|k\rangle$ states, shown in the first line of equation A.1. There, the sum over $2^n$ is split up into multiple sums over $k_i$ where each sum goes from 0 to 1 for each qubit, which leads to the replacement of $k$ by its binary representation in the exponent. The sum in the exponent can be converted to a tensor product, because the sum of exponents is equivalent to the product of its exponentials. By reordering the tensor product with the sums for the qubits and by directly replacing the sum with the two basis states for a qubit $|0\rangle$ and $|1\rangle$, it is possible to derive the desired product representation of the QFT. This representation is used in section 2.4.2 for the derivation of the quantum circuit of the QFT for $n$ qubits. [1]

$$
\begin{aligned}
|x\rangle &\to \frac{1}{2^{n/2}} \sum_{k=0}^{2^n-1} e^{-2\pi i x k/2^n} |k\rangle \\
&= \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} e^{-2\pi i x (\sum_{l=1}^{n} k_l 2^{-l})} |k_1 \dots k_n\rangle \\
&= \frac{1}{2^{n/2}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \bigotimes_{l=1}^{n} e^{-2\pi i x k_l 2^{-l}} |k_l\rangle \\
&= \frac{1}{2^{n/2}} \bigotimes_{l=1}^{n} \left[ \sum_{k_l=0}^{1} e^{-2\pi i x k_l 2^{-l}} |k_l\rangle \right] \\
&= \frac{1}{2^{n/2}} \bigotimes_{l=1}^{n} \left[ |0\rangle\, e^{-2\pi i x 2^{-l}} |1\rangle \right] \\
&= \frac{(|0\rangle + e^{-2\pi i 0.x_{n-1}} |1\rangle) \otimes (|0\rangle + e^{-2\pi i 0.x_{n-2} x_{n-1}} |1\rangle) \otimes \cdots \otimes (|0\rangle + e^{-2\pi i 0.x_0 x_1 \cdots x_{n-1}} |1\rangle)}{2^{n/2}}
\end{aligned}
\tag{A.1}
$$

# Appendix B

# $R_{ij}$ matrices for $\mathrm{QFT}_6$

The high-dimensional controlled rotation matrices $R_{ij}$ are constructed via the formalism of high-dimensional gate construction, explained in section 3.1. They are especially easy to construct, since they are diagonal matrices, hence all entries are one except for those with a phase $\omega$, if both qubits, represented by the indices $i$ and $j$, are one. The constructed $R_{ij}$ matrices in the next-neighbour cut can be seen on the next page, in greater resolution than in section 3.3.3. There the notation of $X^{\otimes y} = \bigotimes_{i=0}^{y-1} X$ is adapted to $X^{\otimes y} = \mathbb{1}_y \otimes X = \mathrm{block\_diag}_y(X)$, in order to shorten the equations. By identifying the correct scaling of the indices in the matrices as a function of $i$, $j$ and $N$, it is possible to deduce the general formula for the $R_{ij}$ matrices in the next-neighbour cut, given in equation 3.19.

$$R_{01} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_{16} & \\ & \omega^2\mathbb{1}_{16} \end{pmatrix} \end{pmatrix} \quad R_{02} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_{8} & \\ & \omega^3\mathbb{1}_{8} \end{pmatrix}^{\otimes 2} \end{pmatrix} \quad R_{03} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_{4} & \\ & \omega^4\mathbb{1}_{4} \end{pmatrix}^{\otimes 4} \end{pmatrix} \quad R_{04} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} \mathbb{1}_{2} & \\ & \omega^5\mathbb{1}_{2} \end{pmatrix}^{\otimes 8} \end{pmatrix} \quad R_{05} = \begin{pmatrix} \mathbb{1}_{32} & & \\ & \begin{pmatrix} 1 & \\ & \omega^6 \end{pmatrix}^{\otimes 16} \end{pmatrix}$$

$$R_{12} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} \mathbb{1}_{8} & \\ & \omega^2\mathbb{1}_{8} \end{pmatrix} \end{pmatrix}^{\otimes 2} \quad R_{13} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} \mathbb{1}_{4} & \\ & \omega^3\mathbb{1}_{4} \end{pmatrix}^{\otimes 2} \end{pmatrix}^{\otimes 2} \quad R_{14} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} \mathbb{1}_{2} & \\ & \omega^4\mathbb{1}_{2} \end{pmatrix}^{\otimes 4} \end{pmatrix}^{\otimes 2} \quad R_{15} = \begin{pmatrix} \mathbb{1}_{16} & & \\ & \begin{pmatrix} 1 & \\ & \omega^5 \end{pmatrix}^{\otimes 8} \end{pmatrix}^{\otimes 2}$$

$$R_{23} = \begin{pmatrix} \mathbb{1}_{8} & & \\ & \begin{pmatrix} \mathbb{1}_{4} & \\ & \omega^2\mathbb{1}_{4} \end{pmatrix} \end{pmatrix}^{\otimes 4} \quad R_{24} = \begin{pmatrix} \mathbb{1}_{8} & & \\ & \begin{pmatrix} \mathbb{1}_{2} & \\ & \omega^3\mathbb{1}_{2} \end{pmatrix}^{\otimes 2} \end{pmatrix}^{\otimes 4} \quad R_{25} = \begin{pmatrix} \mathbb{1}_{8} & & \\ & \begin{pmatrix} 1 & \\ & \omega^4 \end{pmatrix}^{\otimes 4} \end{pmatrix}^{\otimes 4}$$

$$R_{34} = \begin{pmatrix} \mathbb{1}_{4} & & \\ & \begin{pmatrix} \mathbb{1}_{2} & \\ & \omega^2\mathbb{1}_{2} \end{pmatrix} \end{pmatrix}^{\otimes 8} \quad R_{35} = \begin{pmatrix} \mathbb{1}_{4} & & \\ & \begin{pmatrix} 1 & \\ & \omega^3 \end{pmatrix}^{\otimes 2} \end{pmatrix}^{\otimes 8}$$

$$R_{45} = \begin{pmatrix} \mathbb{1}_{2} & & \\ & \begin{pmatrix} 1 & \\ & \omega^2 \end{pmatrix} \end{pmatrix}^{\otimes 16}$$

Figure B.1: High-dimensional controlled rotation matrices $R_{ij}$ used in the $\mathrm{QFT}_6$

# Appendix C

# Code for numeric prove of $\mathrm{QFT}_N$

In the following, the code in python 3.12 is given to prove numerically the equivalence of the constructed $\mathrm{QFT}_N$ matrix, which consists of high-dimensional Hadamard and controlled rotation matrices and a SWAP gate, and the $\mathrm{DFT}_N$ matrix, which is also shown in equation 3.25. The `gen_U(self, i, j)` and `gen_U_vec(self, i, j)` functions generate the $R_{ij}$ matrices as matrices and vectors, respectively. The `gen_H(self, i)`, `gen_swap(self)` and the `gen_F(self)` functions generate the Hadamard, SWAP and DFT matrix respectively. The functions for the generation of the $\mathrm{QFT}_N$ matrix are given by `gen_QFT(self)` and `gen_QFT_vec(self)`, where the only the difference of these functions is, that the first one uses standard matrix multiplication of the $R_{ij}$ matrices, while the latter calculates the element-wise dot product of the diagonal vectors of $R_{ij}$ first, embeds the result in a diagonal matrix and uses this matrix for further calculation of the $\mathrm{QFT}_N$.

```
1  import numpy as np
2  import math
3  import time
4
5  class QFT:
6
7      def __init__(self, N):
8          self.had2 = np.array([[1, 1], [1, -1]])/np.sqrt(2)
9          self.N = N
10         self.dim = 2**N
11
12     def setN(self, N):
13         self.N = N
14         self.dim = 2**N
15
16     def gen_U(self, i, j):
```

```
17          m = j - i + 1
18          k = self.N - j - 1
19          l = self.N - i - 1
20          id_2k = np.eye(2**k)
21          id_2l = np.eye(2**l)
22          omega = (math.e**(2*math.pi*1j/(2**m)))
23          B_mk = np.block([[id_2k,
               np.zeros_like(id_2k)],[np.zeros_like(id_2k),
               omega*id_2k]])
24          R_mk = np.kron(np.eye(2**(m-2)), B_mk)
25          CR_mkl = np.block([[id_2l,
               np.zeros_like(id_2l)],[np.zeros_like(id_2l), R_mk]])
26          U_ij = np.kron(np.eye(2**i), CR_mkl)
27          return U_ij
28
29      def gen_U_vec(self, i, j):
30          m = j - i + 1
31          k = self.N - j - 1
32          l = self.N - i - 1
33          id_2k = np.ones(2**k)
34          id_2l = np.ones(2**l)
35          omega = (math.e**(2*math.pi*1j/(2**m)))
36
37          B_mk = np.concatenate([id_2k, omega*id_2k])
38          R_mk = np.kron(np.ones(2**(m-2)), B_mk)
39          CR_mkl = np.concatenate([id_2l, R_mk])
40          U_ij = np.kron(np.ones(2**i), CR_mkl)
41          return U_ij
42
43      def gen_H(self, i):
44          H = 1
45          for j in range(i):
46              H = np.kron(H, np.eye(2))
47          H = np.kron(H, self.had2)
48          for j in range(self.N - i - 1):
49              H = np.kron(H, np.eye(2))
50          return H
51
52      def gen_QFT(self):
53          qft = np.eye(self.dim)
54          for i in range(self.N-1, -1, -1):
55              for j in range(self.N-1, i, -1):
56                  qft = qft @ self.gen_U(i, j)
```

```
57              qft = qft @ self.gen_H(i)
58          qft = self.gen_swap() @ qft
59          return qft
60
61      def gen_QFT_vec(self):
62          qft = np.eye(self.dim)
63          for i in range(self.N-1, -1, -1):
64              u_comp = np.ones(self.dim)
65              for j in range(self.N-1, i, -1):
66                  u_comp = u_comp * self.gen_U_vec(i, j)
67              qft = qft @ np.diag(u_comp)
68              qft = qft @ self.gen_H(i)
69          qft = self.gen_swap() @ qft
70          return qft
71
72      def gen_swap(self):
73          size = 2**self.N
74          permutation_matrix = np.zeros((size, size), dtype=int)
75          for i in range(size):
76              binary = np.array(list(np.binary_repr(i,
                    width=self.N)), dtype=int)
77              for j in range(self.N // 2):
78                  left = j
79                  right = self.N - 1 - j
80                  binary[left], binary[right] = binary[right],
                        binary[left]
81              new_index = int("".join(binary.astype(str)), 2)
82              permutation_matrix[new_index, i] = 1
83          return permutation_matrix
84
85      def gen_F(self):
86          F = np.zeros((self.dim, self.dim), dtype=complex)
87          for i in range(self.dim):
88              for j in range(self.dim):
89                  F[i, j] =
                        (math.e**(2*math.pi*1j/self.dim))**(i*j)
90          return F/np.sqrt(self.dim)
91
92
93  # Testing correctness
94  qft = QFT(14)
95  start = time.time()
96  res = qft.gen_QFT_vec()
```

```
97 stop = time.time()
98 F = qft.gen_F()
99 correct = np.allclose(F, res)
100 time = np.round((stop - start, 6))
```