Vrije Universiteit in Amsterdam
Faculty of Sciences
Division of Mathematics and Computer Science
Department of Theoretical Computer Science

MASTER'S THESIS

Mgr. Robert Špalek

Quantum Circuits with unbounded fan-out

Supervisor: Prof. dr. Harry Buhrman, CWI
Second reader: dr. Femke van Raamsdonk, VU

Amsterdam 2002

Acknowledgements

First of all, I would like to thank my supervisor Prof. dr. Harry Buhrman from the Centrum voor Wiskunde en Informatica (CWI) in Amsterdam for all the help and his time he spent with me. I also thank him for his useful remarks regarding the thesis and many discussions about theoretical problems.

Furthermore, I would like to thank Dipl.-inform. Hein Röhrig and dr. Hartmut Klauck from CWI, and Prof. Frederic Green from Clark University in Worcester for taking part in these discussions and for their support that helped me a lot. I would like to thank my second reader dr. Femke van Raamsdonk from the Vrije Universiteit in Amsterdam for her support concerning my studies at the Vrije Universiteit, and to the supervisor of my Czech thesis RNDr. Pavel Pudlák, DrSc. from Academy of Sciences in Prague for introducing me to Quantum Computation.

I also thank to the Vrije Universiteit in Amsterdam for admitting me to the One Year Master's Program 2001/2002, inviting me to Amsterdam, and feeding me for the whole school year.

Last but not least, I want to thank Ája for her patience, to all my friends, and most of all to my parents for the support they gave me during my studies.

Amsterdam, July 17, 2002 Mgr. Robert Špalek

Contents

1 Preliminaries **1**
- 1.1 Contents of the thesis . . . 1
- 1.2 Contribution of the thesis . . . 2
- 1.3 Notation . . . 3

2 Quantum Computation **5**
- 2.1 Intrinsics of Quantum Computation . . . 5
  - 2.1.1 State space . . . 5
  - 2.1.2 Evolution . . . 7
  - 2.1.3 Measurement . . . 9
- 2.2 Models of Quantum Computation . . . 11
  - 2.2.1 Quantum Circuits . . . 12
  - 2.2.2 Quantum Turing Machines . . . 12
  - 2.2.3 Quantum Branching Programs . . . 12
- 2.3 Approximations . . . 12

3 Quantum Circuits with fan-out **15**
- 3.1 Survey of existing circuit classes . . . 15
- 3.2 Motivation . . . 16
- 3.3 Quantum fan-out operation . . . 16
- 3.4 Quantum parity operation . . . 18
- 3.5 Generalised quantum operations . . . 20

4 Parallelisation method **23**
- 4.1 Parallelising commuting operations . . . 23
- 4.2 Application to one qubit operators . . . 27

5 Quantum Fourier Transform **29**
- 5.1 Definition and decomposition . . . 29
- 5.2 Basic narrow circuit . . . 31
- 5.3 Decomposition of QFT . . . 32

5.4 Parallelising QFS 33
5.5 Implementing QFP 34
5.6 Precision of the approximate QFT 37

6 Increment operation **41**
6.1 Classical circuit 41
6.2 Quantum circuit 43

7 Interesting shallow circuits **45**
7.1 Approximative Counting and Threshold gate 46
7.2 Approximative Rank and Toffoli gate 48
7.3 Exact Rank and Toffoli gate 49
7.4 Exact Threshold and Counting gate 52
7.5 Arithmetics 54

8 Lower and upper bounds **55**
8.1 Arithmetics using threshold circuits 55
8.2 Factoring problem 57
8.3 Relations of the quantum circuit classes 59
8.4 Open problems 60

List of Figures

2.1 Bloch sphere representation of a qubit 7

3.1 Design of a Quantum Circuit with unbounded fan-out . . . 17
3.2 Equivalence of the fan-out and parity gates 19
3.3 Implementing an arbitrary controlled one qubit operation . 22

4.1 A serial circuit performing $U = \prod_{i=1}^{n} U_i^{x_i}$ 23
4.2 A serial circuit with interpolated changes of basis 24
4.3 A parallelised circuit performing $U = T^{\dagger}(\prod_{i=1}^{n} V_i^{x_i})T$ 25

5.1 The basic narrow circuit performing the QFT 31
5.2 The parallelised shallow circuit computing an output qubit of the QFS 33

6.1 Computing logical conjunctions of all prefixes $c_k = \bigwedge_{j=0}^{k} x_k$. 42
6.2 Diagonal form $D = FPF^{\dagger}$ of the increment operator P 44

7.1 A serial circuit implementing the linear Counting gate 46
7.2 A parallel circuit implementing the linear Threshold gate . . 48
7.3 The Rank gate consists of an iterated $n \to \mathrm{O}(\log n)$ reduction 51
7.4 The Counting gate built up from Rank gates 54

8.1 Relations of the circuit classes 59

Chapter 1

Preliminaries

1.1 Contents of the thesis

We propose a new quantum circuit model — Quantum Circuits with unbounded fan-out. We investigate its power, properties, and relations to existing circuit models. The document has the following structure:

Chapter 2 is an introductory chapter. We introduce the notions of quantum computing. A very short recapitulation of quantum matters (state space, evolution, measurement) is included, though this document does not serve as an introduction course of quantum computing. We mention various quantum models of computation: Quantum Circuits, Quantum Turing Machines, and Quantum Branching Programs. We also define approximations of unitary operations and prove a simple theorem about them.

In Chapter 3, we propose the desired model. We define the quantum equivalent of the fan-out operation and the quantum parity operation. We show that these two operations are indeed equivalent in power. Furthermore, we generalise these operations to controlled one qubit gates. We prove that the power of this model in unchanged, if we allow an arbitrary number of fan-out and parity gates operating on the same qubits in one layer (assuming that we do not mix source and target qubits).

Chapter 4 contains the description of a general parallelisation method. It can parallelise applying an arbitrary number of commutative gates on the same qubits. The price of this speedup is bigger space complexity. The intrinsics of this method are: finding a basis, in which the operators are diagonal, converting the operators into this basis, and incorporating the quantum fan-out operation.

Chapter 5 deals with the main tool for many circuit constructions — the Quantum Fourier Transform (QFT). We define it and find a decomposition that leads to efficient circuits. We outline a basic circuit computing the QFT in place (without ancilla qubits). Since this circuit is quite deep, we investigate the decomposition of the QFT into two simpler operations: the QFS and the QFP. We show that the QFS can be implemented exactly in constant depth. However the QFP seems to be much more difficult and we present a logarithmic depth approximation circuit taken from [CW00]. We analyse the precision of the circuits and prove, that good approximate circuits can not be shallow.

Chapter 6 describes an efficient circuit for the integer increment operation. The increment operator is diagonal in the Fourier basis and, in this basis, it can be implemented by a constant depth circuit.

Chapter 7 contains descriptions of several shallow circuits for the Toffoli, Majority, Threshold, Rank, and Counting operations. Using the parallelisation method, the QFT, and the increment operator, we can approximate all of them in depth $O(\log\log n)$ — by simply performing the increment operation controlled by each source qubit (in parallel). We further investigate what happens, if we replace the QFT by the Hadamard transform. We prove that using this method, we yield an exact $O(\log^* n)$ depth circuit for the Rank operation. Furthermore, this circuit can be used to infer $O(\log^* n)$ depth exact circuits also for other operations (at the price of a bigger space complexity). Finally, we deal with arithmetical circuits.

Chapter 8 presents several important related results from other articles ([SBKH93, Sho97]) and applies them to the developed circuit model. We mention, that several arithmetical operations can be implemented by a threshold circuit of constant depth, hence they can be computed by a quantum circuit with fan-out of $O(\log^* n)$ depth. We review Shor's factoring algorithm and show, that the quantum part of the algorithm can be implemented by a quantum circuit of logarithmic depth. This suggests, that QNC^1 is probably not contained in P.

The electronic version of this document can be found on the Internet at URL `http://www.ucw.cz/~robert/qncwf/`, you can also contact the author by e-mail `robert@ucw.cz`.

1.2 Contribution of the thesis

We have designed a new quantum circuit model $\mathrm{QNC}^k_\mathrm{f}$, which is contained in QACC^k (defined in [Moo99]). The difference between that 2 models is, that $\mathrm{QNC}^k_\mathrm{f}$ does not have a Toffoli gate as one of the basic gates.

A simple variant of the parallelisation method has been already proposed in [Moo99]. In this thesis, we generalise the method.

The QFT and its decomposition to QFS and QFP have been investigated in detail in [CW00, NC00]. In this thesis, we implement exactly the QFS by a quantum circuit with fan-out of constant depth (the most shallow exact quantum circuit has logarithmic depth). However it seems, that the quantum fan-out does not help when implementing the QFP, i.e. we have not found a good sub-logarithmic approximate circuit.

We have discovered, that the increment operator in the Fourier basis can be implemented exactly in constant depth. Using this result, the parallelisation method, and the approximate QFT, we obtain several double-logarithmic approximate circuits for the Counting, Threshold, and Rank gates (and their simpler variants: the Toffoli and Majority gates).

We have also developed a new method. We replace the QFT in the circuit for the Rank gate by the Hadamard transform. We show, that we no longer obtain the exact rank as the output of the circuit after the replacement, however it is possible to test reliably whether the rank is equal to a fixed value. There exists an exact constant depth quantum circuit with fan-out reducing the input size n to $\mathrm{O}(\log n)$. By iterating this reduction, we obtain a shallow ($\mathrm{O}(\log^* n)$ depth) and exact circuit for the Rank gate. Since the Threshold gate can be constructed from the Rank gate, it can be also implemented by a shallow circuit.

Using results from [SBKH93], it implies that several arithmetical operations and sorting can be performed in depth $\mathrm{O}(\log^* n)$.

1.3 Notation

As usual, $\mathbf{N}$, $\mathbf{N}_0$, $\mathbf{Z}$, $\mathbf{Z}_n$, $\mathbf{R}$, $\mathbf{R}_0^+$, and $\mathbf{C}$ will denote natural numbers, natural numbers including 0, integers, integer numbers $\{0,1,2,\ldots,n-1\}$, real numbers, non-negative real numbers and complex numbers.

For any set S, both $\#S$ and $|S|$ will denote the number of elements in S and $P(S)$ will denote the power set of S, i.e. the set of all subsets of S. For any set S, S^n is the set of all sequences of S of length n, S^* is the set of all finite sequences of S. For any mapping f and set S, $f(S)$ will denote $\{f(s)|s \in S\}$.

We use the standard notation $g = \mathrm{O}(f)$, $g = \mathrm{o}(f)$, $g = \Omega(f)$, and $g = \Theta(f)$ for the asymptotic relation of two functions f, g.

For any finite or countable set S, $\ell_2(S)$ will denote the Hilbert space whose elements are mappings from S to C. Elements of such spaces will be expressed using Dirac notation; for each $s \in S$, $|s\rangle$ denotes the elementary

unit vector taking value 1 at s and 0 elsewhere, and $\{|s\rangle | s \in S\}$ is the set of base vectors of $\ell_2(S)$. For $|\phi\rangle \in \ell_2(S)$, $\langle\phi|$ denotes the linear functional mapping each $|\psi\rangle \in \ell_2(S)$ to the inner product $\langle\phi|\psi\rangle$ (conjugate-linear in the first coordinate rather than the second). The norm of the vector is defined in usual way as $\| |\phi\rangle \| = \sqrt{\langle\phi|\phi\rangle}$. For a matrix M, M^T denotes the transpose of the matrix, M^* denotes the complex conjugate of the matrix and $M^\dagger = (M^T)^*$ denotes the adjoint of the matrix.

A *log-star* function is a mapping $\log^* : \mathbf{R} \to \mathbf{N}$. The value $\log^*(n) = k$ such, that $\log^{(k)}(n) = \log\log\ldots\log n \leq 0$. It is a very slowly growing function, however going to infinity. A *signum* function is a mapping $\mathrm{sgn} : \mathbf{R} \to \{0,1\}$ such, that $\mathrm{sgn}(x) = 1$ iff $x \geq 0$ and $\mathrm{sgn}(x) = 0$ otherwise. For two n-bit positive integers $x, y \in \mathbf{N}_0$, where $x = x_{n-1}\ldots x_1 x_0$, $x \bullet y$ will denote the *bitwise scalar product* of x and y, i.e. $x \bullet y = \sum_{k=0}^{n-1} x_k y_k$.

In the thesis, $p(n)$, $p_k(n)$, ... always means a polynomial function.

Chapter 2

Quantum Computation

We shall not supply an introduction course of Quantum Computation here. Many splendid books have been published about it, e.g. [NC00, Pre97]. However for purposes of the completeness of this document, we shall briefly remind basic principles, notions, and the notation in this chapter. This chapter has been taken almost unchanged from [Špa02].

2.1 Intrinsics of Quantum Computation

A classical computer is situated in a unique and observable state at every instant of the computation. It is possible to dump its memory into a storage medium, examine or modify it and restart the computation from the interrupted state any times we want. The computation is also deterministic and unless an error occurs, it always yields the same result.

A quantum computer behaves completely else, which we shall remind in this section.

2.1.1 State space

A quantum computer can be situated in a superposition of classical states. Its state is completely described by a *state vector* $|\varphi\rangle \in \ell_2(S)$ of complex amplitudes,[1] where S is the set of classical states, e.g. for an n-qubit computer $S = \{0,1\}^n$ thus $|\varphi\rangle \in \mathbf{C}^{2^n}$. The quantum analogue of a bit is called a *qubit*.

The set of *classical states* of an n-qubit quantum computer is called a *computational basis* and the states are labelled by $|i\rangle, i \in \{0,1,\ldots,2^n-1\}$. We

[1] we shall ignore the notion of mixed states in this thesis, i.e. all states considered will be the pure states

say that the linear combination $\sum_i \alpha_i |\varphi_i\rangle$ is the *superposition* of states $|\varphi_i\rangle$ with the *amplitude* α_i of the state $|\varphi_i\rangle$.

Note 2.1 A state vector must fulfil the *normalisation condition*, which is a quantum analogue of the requirement that the probabilities of distinct events sum to 1. It says that $|\varphi\rangle$ is a unit vector, i.e. $\langle\varphi|\varphi\rangle = 1$. It should also be reminded, that the *global phase* is unobservable for computational purposes, hence we do not distinguish between $|\varphi\rangle$ and $\alpha|\varphi\rangle, |\alpha| = 1$.

Example 2.1 An one qubit computer has two classical states $|0\rangle$ and $|1\rangle$ and it can also be situated in the superposition of these states, e.g. $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$, $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$ or say $\frac{|0\rangle-2i|1\rangle}{\sqrt{5}}$.

Note 2.2 There is a visual way of representing a qubit. A qubit may be situated in a general superposition state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. Since $\| |\psi\rangle \| = 1$, $|\alpha|^2 + |\beta|^2 = 1$ and we can rewrite the equation as

$$|\psi\rangle = e^{i\gamma}\left(\cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle\right),$$

where θ, φ, γ are real numbers. We shall ignore the unobservable global phase $e^{i\gamma}$, hence the qubit state is completely described by two real variables. The two numbers θ, φ define a point on the unit three-dimensional sphere, often called *Bloch sphere*, see Figure 2.1. It provides a useful visualisation of most one qubit quantum operations. Unfortunately there is no direct generalisation to multiple qubits.

Note 2.3 It turns out that the *phase* of an individual quantum state in a superposition state is a quantity of the same importance as the identity of the state itself. This phenomenon is distinct from the fact, that the global phase is unobservable. We can not distinguish between $|0\rangle + |1\rangle$ and $-|0\rangle - |1\rangle$ while distinguishing between $|0\rangle + |1\rangle$ and $|0\rangle - |1\rangle$ is trivial thanks to the Hadamard operation defined in the next subsection.

Note 2.4 Notice that the state space of a joint system is a tensor product of the state spaces of the individual systems. The composite state $|\varphi\rangle|\chi\rangle$ is also denoted by $|\varphi\chi\rangle$.

Nevertheless it is not true that every composite state is a tensor product of the individual states. This phenomenon is called *entanglement* and such individual states are called entangled. There is an extremely important

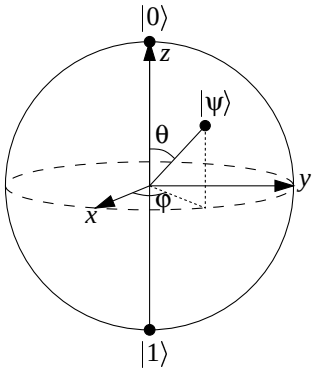

Figure 2.1: Bloch sphere representation of a qubit

example called EPR-pair which serves as a useful tool for many applications (super-dense coding, quantum state teleportation and many others; see [NC00]). One of the four EPR-pairs can be written as

$$|\chi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}},$$

the other three differ by the sign or by flipping the first qubit.

2.1.2 Evolution

A computational step of a quantum computer is described by an *evolution operator*. Quantum physics requires that this operator must be *unitary*, i.e. reversible and norm-preserving. If a quantum computer is situated in the state $|\varphi\rangle$, then it switches to the state $U|\varphi\rangle$ after the operator U is performed.[2] Recall that the product of unitary operators is also a unitary operator.

Example 2.2 Every permutation operator is unitary, hence any classical reversible operation is also permitted in the quantum world. The simplest operators on a qubit are perhaps the *identity* I and the *bit flip* operator X. The operators have the following representations in the computational basis $|0\rangle, |1\rangle$:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

[2] the quantum evolution is indeed a continuous process described by a Hamiltonian, but this fact is not important for our purposes

The proper quantum examples of evolution operators are the *phase flip* operator Z and the combined flip operator Y. The operators I, X, Y, Z form a basis over the space of one qubit operators, they are called *Pauli operators* and they will be mentioned a few times in the document.

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

The last but not the least important operator is the *Hadamard operator H*.

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

The Hadamard operator has a very interesting property. If we apply it to the basis state $|0\rangle$, we obtain $(|0\rangle + |1\rangle)/\sqrt{2}$, which is a *uniform superposition* of either states. If we apply it once more, we obtain the basis state $|0\rangle$ again.

Note 2.5 It is very illuminating to imagine how do the one qubit operators act on the Bloch sphere. The identity I leaves the state unchanged. The Pauli operators X, Y, Z reflect the state through the x, y, z axes respectively. The Hadamard operation H performs a rotation of the sphere about the y axis by 90° followed by a reflection through the $x-y$ plane.

Another interesting examples are the *rotation operators*. A simple algebra shows, that the operator

$$R_x(\theta) = e^{-i\theta X/2} = \cos\frac{\theta}{2} I - i \sin\frac{\theta}{2} X$$

rotates the Bloch sphere about the x axis by angle θ. Similar operators $R_y(\theta)$, $R_z(\theta)$ can be defined also for other axes.

Example 2.3 The *controlled NOT* operator (CNOT) acts on two qubits. It has the following representation in the computational basis $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$:

$$\mathrm{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

If the first qubit is nonzero it does nothing, otherwise it flips the second qubit.

Note 2.6 Quantum physics allows us to apply *any* unitary operation in unit time, at least in principle.[3] As a matter of fact, only a small set of operations is physically feasible. It can be shown (see [NC00]) that there exists a small (discrete) set of operations that forms a *universal set*, i.e. every quantum operation operating on two qubits can be simulated by a finite sequence of the universal operators with the precision exponential in the length of the sequence. Moreover every quantum operation on n qubits can be decomposed into a finite sequence of two qubit operations. This sequence is unfortunately exponentially long for most operators — it can be proved by a simple counting argument.

The goal of the quantum computational complexity is finding which operators can be implemented more efficiently and describing such implementations. This is what the quantum algorithm design is about.

Note 2.7 One of the pitfalls of Quantum Computation is the impossibility of copying unknown quantum states, called *no-cloning theorem*. It is explained in detail in [NC00, page 532] and it claims, that there exists no evolution operator U such, that $U(|\psi\rangle|0\rangle) = |\psi\rangle|\psi\rangle$ for every superposition state $|\psi\rangle$. An operator performing this mapping is inevitably not unitary.

2.1.3 Measurement

From a computer scientists point of view quantum physics provides a powerful tool for a fast multiplication of exponentially large matrices of complex numbers. Though the matrices need to be unitary and easily decomposed, it seems to lead to an exponential speedup over classical computers. However we encounter a substantial problem at the moment — the amplitudes of a quantum state are hidden and protected from direct observations.

The only way how to obtain information from a quantum computer is observing it. The *observation* is described by an *observable*. Every observation inevitably disturbs the quantum state and projects the state vector into some vector subspace. The more information we get by the measurement, the more we disturb the state.

Definition 2.1 An *observable* is a collection $\{M_m\}_m$ of measurement operators. The index m refers to the measurement outcomes that may occur in the experiment. If a quantum computer is situated in the state $|\varphi\rangle$ immediately before the experiment, then the probability that result m occurs is

[3] e.g. the operator *SAT* that solves the satisfiability problem and flips the target bit if the input problem has a solution is a unitary operator

$p(m) = \| M_m|\varphi\rangle \|^2$ and the state of the system after the measurement is

$$\frac{1}{\| M_m|\varphi\rangle \|} M_m|\varphi\rangle.$$

The measurement operators must satisfy the *completeness condition*

$$\sum_m M_m^\dagger M_m = I.$$

Note 2.8 The definition of the observable just stated is very general and allows us to perform so-called *POVM measurements* (see [NC00]). For our purposes, it suffices that the measurement operators are orthogonal projection operators, i.e. it holds that $M_i M_j = \delta_{i,j} M_i$ in addition to the completeness condition. This kind of measurement is called a *projective measurement*. It turns out that if we can perform an additional unitary operation before the measurement, this restricted model is equivalent to the general one.

Example 2.4 Perhaps the simplest projective measurement is the *measurement in the computational basis*. There are two interesting observables of the one qubit system of this type:

$$\begin{aligned} O_{\text{proj}} &= \{|0\rangle\langle 0|, |1\rangle\langle 1|\}, \\ O_{\text{Id}} &= \{|0\rangle\langle 0| + |1\rangle\langle 1|\} = \{I\}. \end{aligned}$$

Let us apply the O_{proj} measurement to a qubit. If the qubit is in the superposition state $\alpha|0\rangle + \beta|1\rangle$ where $|\alpha|^2 + |\beta|^2 = 1$, a simple calculation yields that the probability of observing 0 is $|\alpha|^2$ and the target quantum state after the measurement is $|0\rangle$. Further, the probability of observing 1 is $|\beta|^2$ and the target quantum state after the measurement is $|1\rangle$. On the contrary, the measurement O_{Id} leaves the quantum state untouched and it reveals no information at all.

Composing projective measurement observables of composite systems is straightforward. They can be composed by a tensor multiplication of the individual measurement operators, e.g. if we want to observe only the first qubit of two qubits, we use the observable O_{bit1}, if we want to observe both qubits, we use the observable O_{both}:

$$\begin{aligned} O_{\text{bit1}} &= \{|00\rangle\langle 00| + |10\rangle\langle 10|, |01\rangle\langle 01| + |11\rangle\langle 11|\}, \\ O_{\text{both}} &= \{|00\rangle\langle 00|, |01\rangle\langle 01|, |10\rangle\langle 10|, |11\rangle\langle 11|\}. \end{aligned}$$

O_{both} collapses the two qubit system into a particular basis state. The behaviour of O_{bit1} is more interesting. It collapses the system into a superposition of states consistent with the measurement outcome, leaving their

amplitudes untouched except for rescaling. For example, if the system was in the superposition state $\sum_{i=0}^{3}\alpha_i|i\rangle$ immediately before the measurement and the outcome 1 was measured, then the system will collapse to the state $\frac{1}{|\alpha_1|^2+|\alpha_3|^2}(\alpha_1|01\rangle+\alpha_3|11\rangle)$.

Example 2.5 An important example of an observable in other than computational basis is

$$\begin{aligned} O_{\text{Had}} &= \{\tfrac{1}{2}(|0\rangle+|1\rangle)(\langle 0|+\langle 1|),\tfrac{1}{2}(|0\rangle-|1\rangle)(\langle 0|-\langle 1|)\} = \\ &= \left\{\tfrac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix},\tfrac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}\right\}. \end{aligned}$$

It is nothing else than a projective measurement in the basis $(|0\rangle+|1\rangle)/\sqrt{2}$, $(|0\rangle-|1\rangle)/\sqrt{2}$. We denote it by O_{Had}, because the conversion operator between the computational basis and this one is the Hadamard operator.

We see that measurements reveal very poor information about the original quantum state — the quantum states form a continuum and the measurement outcome is a discrete value. Unfortunately the state is disturbed by the measurement, hence the measurement can not be repeated to reveal more information. The (difficult) task of the quantum complexity theory is developing quantum algorithms that yield the desired information merely from the measurement outcomes.

Note 2.9 One of the essential problems in quantum computing is the problem of distinguishing quantum states. Having a promise that the system is situated in one of the fixed quantum states, our task is determining the quantum state. It turns out that doing this with probability 1 is impossible unless the fixed states are orthogonal. It follows from the completeness condition of the observables. This is also the reason why it is impossible to encode reliably more than 1 classical bit into a qubit — the dimension of the qubit vector space is 2, hence there are only two orthogonal vectors there.

2.2 Models of Quantum Computation

Several models of quantum computation have been proposed. Each of them is suitable for some purposes.

2.2.1 Quantum Circuits

Perhaps the most natural model of quantum computation is a Quantum Circuit model. The basic model has been very well described in [NC00]. The extended model (containing the unbounded quantum fan-out operation) has been investigated in [GHMP02, Moo99]. This thesis is all about quantum circuits, and they are described in Chapter 3.

The quantum circuit model is non-uniform in principle. If we prescribe that the sequence of quantum circuits has to be generated by a Turing Machine, we obtain a uniform version of the model.

2.2.2 Quantum Turing Machines

A Quantum Turing Machine model has been proposed in [Wat98] for the sake of study of quantum space complexity. This model is uniform in principle, however there exists also a non-uniform version (Quantum Turing Machine with advice). We do not discuss this model further in this thesis.

2.2.3 Quantum Branching Programs

A Quantum Branching Program model was described independently by many authors. In [Špa02], there is proved the equivalence with Quantum Turing Machines in both uniform and non-uniform case and the relation to deterministic and probabilistic branching programs. This model has a very little inner structure, hence it is hard to write programs in it. However it is ideal for proving lower bounds. We do not discuss this model further in this thesis.

2.3 Approximations

Let us define some useful measures of how close are two given quantum states. The fidelity and the trace distance are defined (for more general mixed states) and investigated in detail in [NC00]. However, here we limit ourselves to consider pure states only.

Definition 2.2 Let $|\varphi\rangle$ and $|\psi\rangle$ be pure quantum states. The *fidelity* of the states is defined by $\mathbf{F}(|\varphi\rangle,|\psi\rangle) = |\langle\varphi|\psi\rangle|$. The *trace distance* of the states is defined by $\mathbf{D}(|\varphi\rangle,|\psi\rangle) = \sqrt{1-\mathbf{F}(|\varphi\rangle,|\psi\rangle)^2}$. The *Euclidean distance* of the states is defined by $\| |\varphi\rangle - |\psi\rangle \|$.

Note 2.10 The fidelity and the trace distance are numbers in interval $\langle 0,1\rangle$. The trace distance and the Euclidean distance are equivalent metrics, i.e. if one goes to zero, so does the other one. The fidelity can be regarded as a cosinus of the angle between two vectors.

Definition 2.3 We say that unitary operator V *approximates* U *with error* ε iff, for every pure quantum state, the Euclidean distance of applying U to the state and V to the state is at most ε. We measure the distance in the Hilbert space that includes the source / target qubits and the ancilla qubits. We define the *error of approximation* by $E(U,V) = \max_{|\psi\rangle} \| (U-V)|\psi\rangle \|$.

Theorem 2.1 Let V be an approximate operator operating on n qubits, that achieves precision ε for each computational basis state. Then V is an approximate operator with precision $\varepsilon \cdot 2^{n/2}$. The expected precision for a random superposition state stays $O(\varepsilon)$.

Proof. Let $\{|x_k\rangle\}_{k=0}^{2^n-1}$ be the computational basis. The state $|\tilde{y}_k\rangle = V|x_k\rangle$ is an approximation of the desired state $|y_k\rangle$ with precision ε, i.e. $\| |\tilde{y}_k\rangle - |y_k\rangle \| \leq \varepsilon$. Let $|x\rangle = \sum_{k=0}^{2^n-1} \alpha_k |x_k\rangle$ be a general superposition state. Using linearity of unitary operators, the arithmetical-quadratical inequality, and the fact that $\sum_k \alpha_k^2 = 1$, it holds that:

$$\begin{aligned}
\| |\tilde{y}\rangle - |y\rangle \| &= \| \sum_k \alpha_k |\tilde{y}_k\rangle - \sum_k \alpha_k |y_k\rangle \| = \| \sum_k \alpha_k (|\tilde{y}_k\rangle - |y_k\rangle) \| \\
&\leq \sum_k |\alpha_k| \cdot \| |\tilde{y}_k\rangle - |y_k\rangle \| \leq \varepsilon \cdot \sum_k |\alpha_k| = 2^n \varepsilon \cdot \frac{\sum_{k=0}^{2^n-1} |\alpha_k|}{2^n} \\
&\leq 2^n \varepsilon \cdot \sqrt{\frac{\sum_k |\alpha_k|^2}{2^n}} = 2^n \varepsilon \cdot \sqrt{\frac{1}{2^n}} = 2^{n/2} \varepsilon.
\end{aligned}$$

However the expected precision will be also $O(\varepsilon)$. Let X_k be a complex random vector describing the error of the approximation $|\tilde{y}_k\rangle - |y_k\rangle$. Let $X = \sum_{k=0}^{2^n-1} \alpha_k X_k$ be the total error of the approximation. Let us assume that

- all random variables X_k have the same distribution,
- they are independent,
- and their expected value is zero, i.e. $EX_k = 0$.

Since $\| X_k \| \leq \varepsilon$, the variance $\mathrm{var} X_k$ is bounded by $O(\varepsilon^2)$. From Central Limit Theorem, X has approximately normal distribution. It is easy to see, that $EX = \sum_k \alpha_k \cdot EX_k = 0$ and $\mathrm{var} X = \sum_k |\alpha_k|^2 \mathrm{var} X_k = \mathrm{var} X_k$. Let us estimate the expected value of X. Using Chebychev's inequality, $P[\| X - EX \| \geq \lambda \cdot \sqrt{\mathrm{var} X}] \leq 1/\lambda^2$, i.e. $P[\| X \| \geq \lambda\varepsilon] \leq 1/\lambda^2$. □

Chapter 3

Quantum Circuits with fan-out

3.1 Survey of existing circuit classes

There exist four basic models of classical circuits. All models comprise AND, OR, and NOT logical gates and the so-called unbounded fan-out operation.

- NC^k comprises fan-in 2 gates and the circuit has depth $\mathrm{O}\left(\log^k n\right)$,
- AC^k comprises unbounded fan-in gates,
- $\mathrm{ACC}^k[q]$ comprises the modulo q gate in addition, and ACC^k is the union of all such classes $\mathrm{ACC}^k = \bigcup_{q\in\mathbf{N}} \mathrm{ACC}^k[q]$,
- TC^k comprises the unbounded fan-in linear threshold gate (see Definition 7.2).

Obviously $\mathrm{NC}^k \subseteq \mathrm{AC}^k \subseteq \mathrm{ACC}^k[q] \subseteq \mathrm{ACC}^k$ and $\mathrm{AC}^k \subseteq \mathrm{TC}^k$.

The following models of quantum circuits have been previously defined in [Moo99].

- QNC^k comprises arbitrary one qubit gates and the controlled NOT gate,
- QAC^k comprises the general Toffoli gate in addition (the NOT operation controlled by a logical conjunction of unbounded number of source qubits),
- QACC^k comprises the unbounded quantum fan-out gate in addition,
- QTC^k comprises arbitrary one qubit gates and the unbounded fan-in linear threshold gate.

Obviously $\mathrm{QNC}^k \subseteq \mathrm{QAC}^k \subseteq \mathrm{QACC}^k$ and $\mathrm{QAC}^k \subseteq \mathrm{QTC}^k$.

However a model comprising arbitrary one qubit gates and the quantum fan-out gate (and not including explicitly the Toffoli gate) has not yet been explored. We shall investigate it and show its relation to existing quantum circuit models.

3.2 Motivation

Taking into account the unbounded fan-out operation is kind of natural due to the following observation. From a physical framework, it follows, that commuting quantum operations can possibly be performed at the same time — if two quantum operations commute, so do their Hamiltonians and we can perform the joint operation by simply performing either evolutions at the same time.

In particular, the quantum fan-out operation is a bunch of controlled NOT operations operating on the same source qubit. We can imagine that the bunch of target qubits can be evolved at the same time using a shared external power.

It turns out that the capability of performing the quantum fan-out operation in constant depth is strong enough to simulate a general set of commuting operations. If we neglect the change of basis (see Theorem 4.1), then the depth of the parallel circuit is approximately equal to the maximal depth of an operation. This is an argument for the inner consistency of the model.

3.3 Quantum fan-out operation

First of all, notice that quantum circuits can not involve a naive quantum counterpart of the classical fan-out operation performing

$$|\phi\rangle|0\rangle^{\otimes n} \to |\phi\rangle^{\otimes(n+1)}, \tag{3.1}$$

for a general superposition state $|\phi\rangle$. This operation is indeed physically infeasible, because of the *no-cloning theorem* (see [NC00]). Notice, that it is not linear, let alone unitary. However a modified quantum fan-out operation can be defined. It performs exactly (3.1) for each computational basis state $|\phi\rangle \in \{|0\rangle, |1\rangle\}$ and the effect on superposition states $|\phi\rangle$ is determined by linearity. The quantum fan-out operation can be also regarded as a bunch of controlled NOT operations sharing the source qubit.

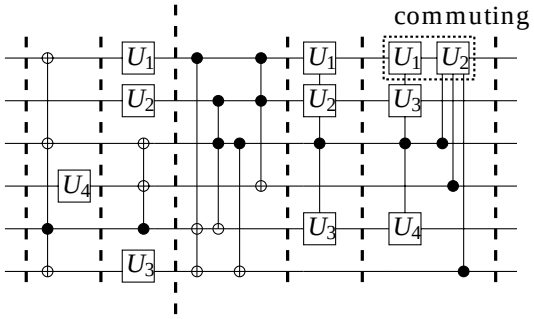

Figure 3.1: Design of a Quantum Circuit with unbounded fan-out

Definition 3.1 A *quantum fan-out operation* on source qubit s and n target qubits t_k performs (written in the computational basis)

$$|s\rangle \bigotimes_{k=1}^{n} |t_k\rangle \rightarrow |s\rangle \bigotimes_{k=1}^{n} |(t_k + s) \bmod 2\rangle$$

for computational basis states and the behaviour for superposition states is defined by linearity.

Definition 3.2 A *Quantum Circuit with unbounded fan-out* is a quantum circuit consisting of arbitrary one qubit gates and quantum fan-out gates. The fan-out gate of arbitrary size is regarded as a unit depth element.

Definition 3.3 The *depth* of the circuit is defined recursively in usual way: Input qubits are considered to have depth 0. For each gate G, the depth of G is equal to 1 plus the maximal depth of a gate G depends on. The depth of the circuit is equal to the maximal depth of a gate. We define the *size* of the circuit to be the total number of gates.

Quantum circuits of depth d operating on s qubits can be represented by a rectangular grid having s horizontal lines (representing states of the qubits) and d consecutive columns (layers). In each layer, we can either perform an arbitrary one qubit operation on a qubit or perform a quantum fan-out operation on a subset of qubits. However each qubit takes part in at most one operation in a layer. Look at the first two layers of the example in Figure 3.1.

In the beginning of the computation, all qubits except for the input ones are initialised to $|0\rangle$. The computation proceeds in computational

steps corresponding to the layers of the circuit. At the end of the computation, the qubits are observed in the computational basis. The values of the qubits denote the output of the problem. For decision problems, an input is accepted / rejected according to the value of some qubit.

Definition 3.4 We define $\mathrm{QNC}_{\mathrm{f}}^{k}$ to be the class of problems solvable by families of Quantum Circuits with unbounded fan-out of depth $\mathrm{O}\left(\log^{k} n\right)$ and size polynomial in n, where n is the input size. We also define $\mathrm{QNC}_{\mathrm{f}} = \bigcup_{k\in\mathbf{N}} \mathrm{QNC}_{\mathrm{f}}^{k}$.

Note 3.1 Every quantum operation of a fixed size n can be decomposed into a sequence of arbitrary one qubit operations and controlled NOT operations (which is a special case of the fan-out operation) of length $\mathrm{O}\left(n^{3}4^{n}\right)$ using the methods described in [BBC+95]. Hence the power of $\mathrm{QNC}_{\mathrm{f}}^{k}$ is unchanged if we allow these operations in addition (for a fixed n). Especially the following operations are useful: the fan-in 2 Toffoli gate (the controlled controlled NOT computing a logical AND), and an arbitrary controlled one qubit operation.

3.4 Quantum parity operation

Let us investigate a natural counterpart of the quantum fan-out operation. It is also a bunch of controlled NOT operations, but it shares the target qubit. Having the target qubit initialised to $|0\rangle$, it computes the parity of $x_1, \ldots, x_n$.

Definition 3.5 A *quantum parity operation* on n source qubits s_k and target qubit t performs (written in the computational basis)

$$|t\rangle \bigotimes_{k=1}^{n} |s_k\rangle \to |(t + \sum_{k=1}^{n} s_k) \bmod 2\rangle \bigotimes_{k=1}^{n} |s_k\rangle$$

for computational basis states and the behaviour for superposition states is defined by linearity.

Theorem 3.1 [Moo99] The parity operation is equivalent to the fan-out operation — if one of them can be performed in constant depth, so can the other one.

Proof. First of all, let us state a remarkable relation — if we precede and succeed the controlled NOT operation by Hadamard operations on both

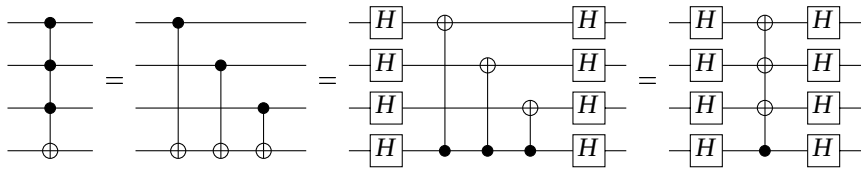

Figure 3.2: Equivalence of the fan-out and parity gates

qubits, the direction of the controlled NOT operation is reverted. Let H denote the Hadamard operation and $C_{k,l}$ denote the NOT operation performed on l-th qubit controlled by the k-th qubit. It holds that

$$(H \otimes H) \cdot C_{0,1} \cdot (H \otimes H) = C_{1,0}.$$

This equation can be verified by simply multiplying the matrix representations of the operators (in the computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, where bits are numbered in the order $|x\rangle = |x_1 x_0\rangle$):

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \qquad H \otimes H = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix},$$

$$C_{0,1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \qquad C_{1,0} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

$$C_{0,1} \cdot (H \otimes H) = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \end{pmatrix}$$

$$(H \otimes H) \cdot C_{0,1} \cdot (H \otimes H) = \frac{1}{4} \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 4 \\ 0 & 0 & 4 & 0 \end{pmatrix} = C_{1,0}.$$

Let us implement the parity operation in constant depth using the fan-out operation and Hadamard operations. The converse simulation is similar. The progress is sketched in Figure 3.2. We first decompose the parity operation into a bunch of controlled NOT operations. We then revert all of them and precede and succeed them by Hadamard operations, which leaves the global operation unchanged. Since two consecutive Hadamard

operations cancel, we obtain a new bunch of controlled operations forming the fan-out operation. The overhead of this simulation is just adding two layers. □

Note 3.2 From the equivalence of the fan-out and parity gates, it is not important which one is included in the definition. The $\mathrm{QNC}^k_\mathrm{f}$ class can be regarded as QNC^k with the unbounded fan-out gate or equivalently as QNC^k with the unbounded parity gate.

3.5 Generalised quantum operations

In this section, we shall outline that even if we allow somehow stronger primitive operations, we would not obtain a stronger model. That is, every circuit using the stronger operations can be simulated by a quantum circuit with fan-out at low cost.

Theorem 3.2 The power of $\mathrm{QNC}^k_\mathrm{f}$ is unchanged if we allow an arbitrary number of fan-out and parity operations on a qubit in a layer (like those in the 3rd layer of the example in Figure 3.1). With the only restriction, that each qubit is either always a source qubit or always a target qubit.

Let S be the set of source qubits and T be the set of target qubits in the layer. Let $n = \#S + \#T$. The layer can be then simulated by a quantum circuit with fan-out having 5 layers and using $\mathrm{O}\left(n^2\right)$ ancilla qubits.

Proof. We first rewrite quantum operations in the layer in such a way that each target qubit is modified by at most one operation. For each target qubit $t \in T$, we compose a parity operation equivalent to the set of original operations concerning it. This is always possible since the total effect of applying more parity and fan-out operations on the same target qubit is always another parity operation. We obtain $\#T = \mathrm{O}(n)$ optimised parity gates. The simulation then works in this way:

1. A fan-out operation of appropriate size is performed on each source qubit $s \in S$ (the number of target qubits t_s is equal to the number of parity operations the qubit s controls). This is the place where the $\sum_{s\in S} t_s \leq \sum_{s\in S} \#T = \#S \cdot \#T = \mathrm{O}\left(n^2\right)$ ancilla qubits are allocated (and initialised to $|0\rangle$).

2. The optimised parity gates are then applied on target qubits T. The source qubits of these gates will not overlap, since we have fanned them out. Hence they can be performed in parallel. Recall that this

indeed costs 3 layers since the parity operation is simulated by the fan-out operation and Hadamard operations.

3. Ancilla qubits are erased by applying the fan-out operations again. This clears the ancilla qubits for a future use, since the fan-out operation is its own inversion.

The constructed quantum circuit with fan-out has 5 layers and uses $\mathrm{O}\left(n^2\right)$ ancilla qubits. □

Definition 3.6 Let $\{V_k\}_{k=1}^n$ be arbitrary one qubit operations. A *generalised quantum fan-out operation* on source qubit s and n target qubits t_k is the operator mapping a computational basis input state

$$|s\rangle \bigotimes_{k=1}^{n} |t_k\rangle \to |s\rangle \bigotimes_{k=1}^{n} V_k^s |t_k\rangle.$$

The behaviour for superposition states is defined by linearity.

Definition 3.7 Let $\{V_k\}_{k=1}^n$ be arbitrary commuting one qubit operations. A *generalised quantum parity operation* on n source qubits s_k and target qubit t is the operator mapping a computational basis input state

$$|t\rangle \bigotimes_{k=1}^{n} |s_k\rangle \to \left(\prod_{k=1}^{n} V_k^{s_k} \right) |t\rangle \otimes \bigotimes_{k=1}^{n} |s_k\rangle.$$

The behaviour for superposition states is defined by linearity.

Note 3.3 The original quantum fan-out / parity operations are just special cases of these generalised operations. It suffices to use the bit flip X as V_k.

Theorem 3.3 The power of $\mathrm{QNC}_{\mathrm{f}}^k$ is unchanged if we allow the generalised quantum fan-out operation (like that in the 4th layer of the example in Figure 3.1). Every such operation can be simulated by a quantum circuit with fan-out having 5 layers and using no additional ancilla qubits.

Proof. It is well known that every controlled one qubit operation U can be decomposed into a circuit consisting of one qubit operations A, B, and C, a phase change P, and two controlled NOT operations. The circuit is outlined in Figure 3.3 and examined in [NC00].

If we are to implement the generalised fan-out operation, we simply repeat the bottom line for each target qubit. Either controlled NOT operations will be replaced by quantum fan-out operations. The phase changes multiply into $P = P_1 \cdots P_n$. □

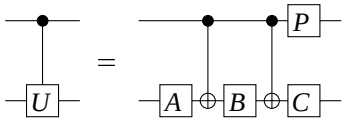

Figure 3.3: Implementing an arbitrary controlled one qubit operation

Note 3.4 We have demonstrated the implementation of the generalised fan-out operation. There exists also an implementation of the generalised parity operation. Moreover, both these generalised quantum operations can be also parallelised at low cost under the restriction described in Theorem 3.2, assuming that the one qubit operations V_k commute — like at the 5th layer of the example in Figure 3.1.

However performing more general operations on a qubit in the same layer is not so straightforward as composing parity operations, hence a more sophisticated parallelisation method is needed. We shall demonstrate it in Theorem 4.3.

Using Theorem 4.3, let us formulate an equivalent definition of Quantum Circuits with unbounded fan-out.

Definition 3.8 A *Quantum Circuit with unbounded fan-out* is a quantum circuit consisting of layers. In each layer, an arbitrary number of generalised fan-out and parity operations can be performed. It must hold, that each qubit is either always a source qubit or always a target qubit in a layer. Furthermore, unitary operations performed on a target qubit in a layer must commute.

Chapter 4

Parallelisation method

In this chapter, we shall show that the capability of performing the unbounded quantum fan-out operation in constant depth is sufficient for being able to perform a much more general task: an arbitrary number of (possibly controlled) commuting operations operating on the same qubits can be performed at the same time in the model of Quantum Circuits with unbounded fan-out at the price of bigger space complexity.

4.1 Parallelising commuting operations

Let us have n commuting unitary operators $U_1, U_2, \ldots, U_n$ operating on k qubits, i.e. $U_i U_j = U_j U_i$ for every i, j. We also have n other control qubits such, that qubit x_i controls, whether U_i is performed. We want to apply all given operators on target qubits. An obvious way is applying them in serial, as shown in Figure 4.1. However since the operators commute, there exists a smarter circuit applying them in parallel.

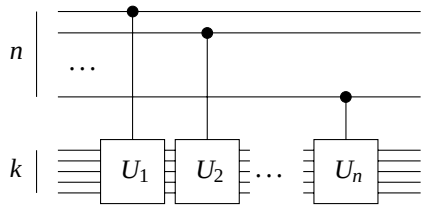

Figure 4.1: A serial circuit performing $U = \prod_{i=1}^{n} U_i^{x_i}$

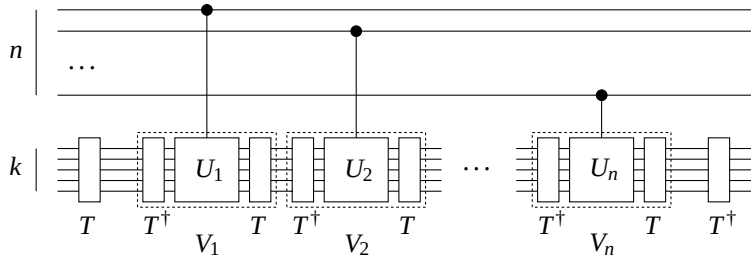

Figure 4.2: A serial circuit with interpolated changes of basis

It is well known that if two operators commute, then they are diagonal in the same basis (see Theorem 4.2). Let $D = \{|d_j\rangle\}_{j=0}^{2^k-1}$ be a basis in which each U_i is diagonal, i.e. $U_i = \sum_{j=0}^{2^k-1} u_{i,j}|d_j\rangle\langle d_j|$, and $B = \{|j\rangle\}_{j=0}^{2^k-1}$ be the computational basis. Let $T = \sum_{j=0}^{2^k-1} |j\rangle\langle d_j|$ be the conversion operator from D to B. Then the operator $V_i = TU_iT^\dagger$ is unitary and diagonal in the computational basis, since

$$V_i|j\rangle = TU_iT^\dagger|j\rangle = TU_i|d_j\rangle = u_{i,j}T|d_j\rangle = u_{i,j}|j\rangle.$$

Since T and $T^\dagger$ are inverses of each other, we can convert the circuit from Figure 4.1 into an equivalent circuit shown in Figure 4.2. It is obvious, that $TU_i^{x_i}T^\dagger = V_i^{x_i}$. Notice, that the latter circuit can be regarded as another instance of the former one — it is also a sequence of controlled operations V_i having the target qubits pre-processed and post-processed. However the operators V_i used now are diagonal in the computational basis.

A diagonal unitary operator consists just of phase shifts, because every coefficient in the diagonal is a complex unit. Furthermore, applying multiple phase shifts on a qubit can be parallelised by using the following trick:

1. Apply the fan-out operation on a qubit to “copy” the state.

2. Apply each phase shift on a distinct “copy”.

3. By applying the fan-out operation again, clear the ancilla qubits.

The intrinsics of this method is the fact that the phase shifts multiply. At one moment, the amplitude of every computational basis state in the superposition is multiplied by all phase shifts of the individual qubits.

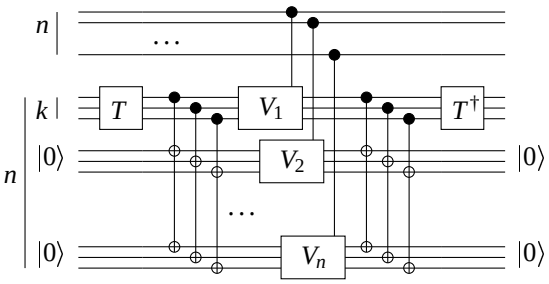

Figure 4.3: A parallelised circuit performing $U = T^\dagger(\prod_{i=1}^n V_i^{x_i})T$

This method can be readily modified for a bunch of qubits. We first copy all target qubits using the fan-out operation, then apply the operators V_i (which perform exactly phase shifts, because they are diagonal in the same basis as the fan-out operation works in, i.e. the computational basis), and finally clear the ancilla qubits. A circuit doing this is shown in Figure 4.3.

Let us prove the correctness of the method. In the beginning, the target qubits are in a general state $|\psi\rangle = \sum_{j=0}^{2^k-1} \alpha_j |j\rangle$ and the ancilla qubits are cleared in the basis state $|0\rangle$. Let us suppose w.l.o.g. the control qubits x_i are not in a superposition. Then the correctness for superposition states follows from linearity. The states of the system at individual stages of the computation are:

$$\begin{aligned}
& |\psi\rangle \otimes |0\rangle^{\otimes(n-1)} \\
= \;& \textstyle\sum_j \alpha_j |j\rangle \otimes |0\rangle^{\otimes(n-1)} \\
\text{fan-out} \to \;& \textstyle\sum_j \alpha_j |j\rangle^{\otimes n} \\
= \;& \textstyle\sum_j \alpha_j \bigotimes_{i=1}^n |j\rangle \\
V_i \text{ performed} \to \;& \textstyle\sum_j \alpha_j \bigotimes_i V_i^{x_i} |j\rangle \\
(V_i \text{ is diagonal}) = \;& \textstyle\sum_j \alpha_j \bigotimes_i v_{i,j}^{x_i} |j\rangle \\
(\text{phase shifts multiply}) = \;& \textstyle\sum_j \alpha_j \left(\prod_i v_{i,j}^{x_i}\right) \bigotimes_i |j\rangle \\
= \;& \textstyle\sum_j \alpha_j \left(\prod_i v_{i,j}^{x_i}\right) |j\rangle^{\otimes n} \\
\text{fan-out} \to \;& \textstyle\left(\sum_j \alpha_j \left(\prod_i v_{i,j}^{x_i}\right) |j\rangle\right) \otimes |0\rangle^{\otimes(n-1)} \\
(V_i \text{ is diagonal}) = \;& \textstyle\left(\sum_j \alpha_j \left(\prod_i V_i^{x_i}\right) |j\rangle\right) \otimes |0\rangle^{\otimes(n-1)} \\
(\text{moving out}) = \;& \textstyle\left(\prod_i V_i^{x_i}\right) \sum_j \alpha_j |j\rangle \otimes |0\rangle^{\otimes(n-1)} \\
= \;& \textstyle\left(\prod_i V_i^{x_i}\right) |\psi\rangle \otimes |0\rangle^{\otimes(n-1)}.
\end{aligned}$$

We see that not only the product of V_i's has been performed, but also the ancilla qubits have been cleared and thus they can be reused.

Theorem 4.1 Let $U_1, U_2, \ldots, U_n$ be commuting unitary operators operating on k qubits. Each operation U_i can be controlled by another qubit x_i. Let D be a basis in which all the operators are diagonal. Let T be a circuit performing a change of basis from D to the computational basis. Then there exists a quantum circuit U with fan-out performing $\prod_{i=1}^{n} U_i^{x_i}$ having depth

$$\mathrm{depth}(U) = \max_{i=1,2,\ldots,n} \mathrm{depth}(U_i) + 4 \cdot \mathrm{depth}(T) + 2$$

and using $(n-1)k$ ancilla qubits.

Proof. Straightforward using the method described above. The depth of the parallelised sub-circuit V is equal to the depth of the deepest V_i plus 2 for the fan-outs. V_i is built from the circuit performing U_i and the circuits $T, T^\dagger$ changing the basis. The desired circuit U consists of V and the circuits $T, T^\dagger$ again. □

A simpler version of this method has been published in [Moo99]. They considered equal operators $U_1 = U_2 = \ldots = U_n$, hence they did not arrive to the restriction of commutativity. Finding a basis, in which U_1 is diagonal, and changing the basis come from that article. The generalisation is original.

For clarity, it remains to show the existence of a proper basis D.

Theorem 4.2 Let $U = \{U_1, U_2, \ldots, U_n\}$ be a set of commuting unitary operators operating on a vector space V. Then there exists a basis $D = \{|d_j\rangle\}_j$ of V such, that all the operators are diagonal in D, i.e. $U_i = \sum_j u_{i,j} |d_j\rangle\langle d_j|$.

Proof. We shall constructively form the basis D using mathematical induction.

Let $e(i)$ be the number of distinct eigen-values of U_i, $\lambda_{i,k}$ be the k-th eigen-value of U_i, $v(i,k)$ be the dimension of the vector space of eigen-vectors corresponding to the eigen-value $\lambda_{i,k}$, and $|v_{i,k,l}\rangle$ be the corresponding eigen-vector. Let $L_{i,k} = L(\{|v_{i,k,l}\rangle\}_{l=1}^{v(i,k)})$ be the vector subspace spanned by eigen-vectors of U_i corresponding to the eigen-value $\lambda_{i,k}$. Notice that every vector in $L_{i,k}$ is also an eigen-vector of U_i.

We have the following task: getting a vector space V and a set of unitary operators U, we are to return a basis D of V such, that each $U_i \in U$ is diagonal in the basis D, i.e. the elements of D are eigen-vectors of each U_i. Let $n = \#U$.

i. If $n = 0$, find any basis of V (using standard methods) and return it.

ii. If $n > 0$, we first decompose U_n on its eigen-values:

$$U_n = \sum_{k=1}^{e(n)} \lambda_{n,k} \left(\sum_{l=1}^{v(n,k)} |v_{n,k,l}\rangle\langle v_{n,k,l}| \right).$$

From the commutativity $(\forall i)\ U_n U_i = U_i U_n$, it holds that

$$\begin{aligned} (\forall i) \quad & U_n(U_i|v_{n,k,l}\rangle) = U_i U_n |v_{n,k,l}\rangle = \lambda_{n,k}(U_i|v_{n,k,l}\rangle) \\ \Longrightarrow \quad & U_i|v_{n,k,l}\rangle \in L_{n,k} \\ \Longrightarrow \quad & U_i\left(L_{n,k}\right) \subseteq L_{n,k}. \end{aligned}$$

(From the unitarity of U_i, the last inclusion is indeed an equality.)

We have decomposed the vector space V into a tensor product of orthogonal vector subspaces $V = \bigotimes_{k=1}^{e(n)} L_{n,k}$ in such a way, that $L_{n,k}$ is closed for each operator U_i and every vector $|v\rangle \in L_{n,k}$ is an eigen-vector of U_n.

If there exist bases $D_1, D_2, \ldots, D_{e(n)}$ of these subspaces such that each operator $U_i \in U - \{U_n\}$ is diagonal in all of them (when restricted to the appropriate subspace), then the set $D = \bigcup_{k=1}^{e(n)} D_k$ is the desired base: it is obviously a base and each vector $|d_j\rangle \in D$ is an eigen-vector of each operator U_i. The existence of the smaller bases follows from mathematical induction — since the subspaces are orthogonal, it suffices to find the bases D_k separately.

(A not important, but perhaps illuminating note: in most cases, the subspaces $L_{n,k}$ will degenerate very fast into one dimension and stay such until $n = 0$.) □

A cumbersome proof for the special case of $n = 2$ operators has been published in [NC00, page 77]. This generalisation is original.

Note 4.1 The operators U_i need not be controlled. If U_i is always performed, just replace the control qubit x_i by 1 at proper places. The method and the proof remain the same.

Note 4.2 The operators U_i need not operate on all k target qubits. If U_i operates on a subset of the qubits, it can be simply regarded as a larger operator operating on all k qubits and not touching some of them.

4.2 Application to one qubit operators

In Note 3.4, we have mentioned that allowing overlapping commutative generalised quantum fan-out and parity operations does not change the

power of $\mathrm{QNC}^k_\mathrm{f}$. However we have proved it only for some special cases (in Theorem 3.2 and Theorem 3.3). Let us use the parallelisation method and prove a more general theorem.

Theorem 4.3 Let B be an arbitrary bunch of (possibly overlapping) commutative generalised fan-out and parity operations operating on n qubits. The operations must fulfil the requirement stated in Theorem 3.2, i.e. each qubit is either always a source qubit or always a target qubit.

Let S be the set of source qubits and T be the set of target qubits in the layer. Let $n = \#S + \#T$. Then B can be simulated by a quantum circuit F with fan-out having 9 layers and using $\mathrm{O}\left(n^2\right)$ ancilla qubits. F has no overlapping operations.

Proof. Straightforward from the presented theorems using the following method:

1. For each target qubit $t \in T$, find a basis in which all applied one qubit operations are diagonal. Since we allow performing an arbitrary quantum operation on a qubit in 1 layer, the changes of basis cost 2 layers.

2. Browse the generalised fan-out and parity operations and, for each source qubit $s \in S$, construct a list of one qubit operations controlled by it (if two operations are applied on the same target qubit, we multiply them) of length t_s. There are $\#S = \mathrm{O}(n)$ lists of lengths $t_s \leq \#T = \mathrm{O}(n)$.

3. "Copy" each target qubit using the fan-out operation. This costs 2 layers and uses $\#S \cdot \#T = \mathrm{O}\left(n^2\right)$ ancilla qubits.

4. For each source qubit, perform the generalised fan-out operation on some copy of the target qubits (using the method described in Theorem 3.3). Since we have "copied" the target qubits, we can do this in parallel. It costs 5 layers.

The correctness follows from the parallelisation method. It is slightly simplified, since the change of basis is performed only twice (because the inner diagonal one qubit operation can be performed directly).

The total overhead of the simulation is: constant depth overhead and quadratic width overhead. The simulation does not significantly change the complexity of the circuit according to Definition 3.4. If we allow overlapping commutative generalised quantum operations in one layer, we do not change the power of $\mathrm{QNC}^k_\mathrm{f}$. □

Chapter 5

Quantum Fourier Transform

5.1 Definition and decomposition

Definition 5.1 A *Quantum Fourier Transform* (QFT) on n qubits is the following operator (written in the computational basis):

$$F = \frac{1}{2^{n/2}} \sum_{t=0}^{2^n-1} |t\rangle \sum_{s=0}^{2^n-1} \xi^{ts} \langle s|, \tag{5.1}$$

where $\xi = e^{2\pi i/2^n}$ is the 2^n-th complex root of unity.

Lemma 5.1 The inverse QFT $F^\dagger$ is defined by equation (5.1) for $\xi = e^{-2\pi i/2^n}$.

Proof. To verify that $F^\dagger$ is indeed the inverse of F, we check whether $FF^\dagger$ is the identity, i.e. $\langle t|FF^\dagger|s\rangle = \delta_{t,s}$. From the definition, $\langle t|FF^\dagger|s\rangle = 2^{-n}\sum_{k=0}^{2^n-1} \xi^{(t-s)k}$. If $t = s$, then the sum equals $\sum_{k=0}^{2^n-1} \xi^0 = 2^n$. Otherwise

$$\sum_{k=0}^{2^n-1} (\xi^{t-s})^k = \frac{(\xi^{t-s})^{2^n} - 1}{\xi^{t-s} - 1} = \frac{(e^{2\pi i})^{t-s} - 1}{\xi^{t-s} - 1} = \frac{1^{t-s} - 1}{\xi^{t-s} - 1} = 0.$$

□

Note 5.1 The Quantum Fourier Transform is a quantum analogue of the Fourier Transform. It, however, operates on quantum amplitudes of the computational states. As presented, it is not usable for performing the Fourier Transform on a given array of numbers.

The representation of F in the computational basis is indeed the Vandermond matrix of numbers $1, \xi, \xi^2, \ldots, \xi^{2^n-1}$.

Let us infer an efficient method of performing the QFT. Let $|x\rangle$ be a *computational basis state,* i.e. x is *not* in a superposition. We shall show that the output qubits of $F|x\rangle$ are not entangled and each of them can be conveyed by a simple formulae.

Theorem 5.2 [GN96, CEMM98] Let $x_{n-1}\dots x_1x_0$ be the binary representation of x, i.e. qubits are ordered in descending order from the most significant one. Then

$$F|x\rangle = \frac{1}{2^{n/2}}\bigotimes_{b=0}^{n-1}\left(|0\rangle + e^{2\pi i\, x/2^{b+1}}|1\rangle\right) = \bigotimes_{b=0}^{n-1}\frac{|0\rangle + e^{2\pi i\,(0.x_b\dots x_1x_0)}|1\rangle}{\sqrt{2}}. \qquad (5.2)$$

Proof. Since exponents are in the form $2\pi i\cdot z$, only the value $z \bmod 1$ is significant. Furthermore, $x/2^{b+1} \equiv (0.x_b\dots x_1x_0) \pmod 1$. Let $y = y_{n-1}\dots y_1y_0$. Let us compute $\langle y|F|x\rangle$ and compare it with the desired value:

$$\begin{aligned}
\langle y|F|x\rangle &= (\langle y_{n-1}|\dots\langle y_1|\langle y_0|)F|x\rangle = \left(\bigotimes_{b=0}^{n-1}\langle y_{n-1-b}|\right)F|x\rangle \\
&= \frac{1}{2^{n/2}}\left(\bigotimes_{b=0}^{n-1}\langle y_{n-1-b}|\right)\left(\bigotimes_{b=0}^{n-1}\left(|0\rangle + e^{2\pi i\, x/2^{b+1}}|1\rangle\right)\right) \\
&= \frac{1}{2^{n/2}}\prod_{b=0}^{n-1}\exp(2\pi i\cdot y_{n-1-b}\cdot x/2^{b+1}) \\
&= \frac{1}{2^{n/2}}\exp\left(2\pi i\cdot\sum_{b=0}^{n-1} y_{n-1-b}\cdot x/2^{b+1}\right) \qquad (\text{let } b = n-1-c) \\
&= \frac{1}{2^{n/2}}\exp\left(2\pi i\cdot x\cdot\sum_{c=0}^{n-1} y_c/2^{n-c}\right) = \frac{1}{2^{n/2}}\exp\left(2\pi i\cdot x/2^n\cdot\sum_{c=0}^{n-1}2^c y_c\right) \\
&= \frac{1}{2^{n/2}}\exp(2\pi i\cdot yx/2^n) = \frac{\xi^{yx}}{2^{n/2}},
\end{aligned}$$

which is the desired value. □

Having inferred this remarkable result, there have been constructed several interesting circuits performing the QFT. The space complexity can be traded-off with the depth of the circuit and with the precision achieved.

Henceforth, let $|\mu_{0.x_k\dots x_1x_0}\rangle = (|0\rangle + e^{2\pi i\,(0.x_k\dots x_1x_0)}|1\rangle)/\sqrt{2}$.

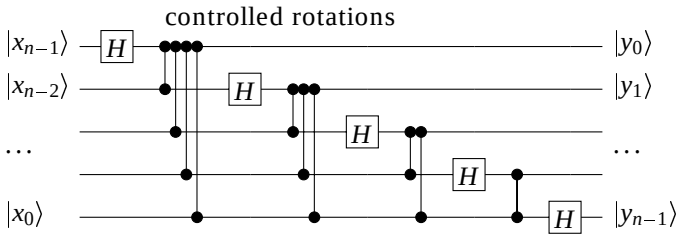

Figure 5.1: The basic narrow circuit performing the QFT

5.2 Basic narrow circuit

Perhaps the best known is the circuit described in [Sho94]. It needs no ancilla qubits, since it operates directly on the source qubits.

Theorem 5.3 The QFT on n qubits can be implemented exactly by a quantum circuit of depth $\mathrm{O}(n^2)$ using no ancilla qubits.

Proof. The circuit is sketched in Figure 5.1. The Hadamard operation turns a computational basis state $|x_b\rangle$ into $(|0\rangle + (-1)^{x_b}|1\rangle)/\sqrt{2} = |\mu_{0.x_b}\rangle$. The two qubit gate operating on the t-th and the s-th qubit is the controlled rotation gate $R_z^c(\pi/2^{|t-s|})$. It is represented in the computational basis by the following matrix:

$$R_z(\alpha) = |0\rangle\langle 0| + e^{i\alpha}|1\rangle\langle 1| = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{pmatrix},$$

$$R_z^c(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\alpha} \end{pmatrix}. \quad \text{(controlled version)}$$

Notice that R_z^c is a symmetric gate, that is why the orientation of the gate is not marked in the figure. Consecutive rotations multiply, hence the overall rotation yielded by the b-th qubit is $e^{2\pi i\,(0.x_b x_{b-1}\ldots x_1 x_0)}$, which is exactly what is needed.

It is obvious that the presented circuit has depth $\mathrm{O}(n^2)$. □

Note 5.2 If the quantum fan-out operation is allowed, the depth of the circuit can be decreased to $\mathrm{O}(n)$. Since the rotations commute, each of the n blocks of controlled rotations can be parallelised into a constant depth circuit. However the parallelisation needs $\mathrm{O}(n^2)$ ancilla qubits.

Let us present also an approximate version of this circuit. It is a circuit where rotations by phases smaller than a given threshold δ are omitted. For each qubit, only $O(\log(1/\delta))$ rotations are performed, hence the depth is linear even in the QNC model. The error of the circuit will be investigated later.

5.3 Decomposition of QFT

Definition 5.2 A *quantum Fourier state computation* (QFS) is any unitary operation mapping (5.3) for each computational basis state $|x\rangle$. A *quantum Fourier phase computation* (QFP) is any unitary operation mapping (5.4) for each computational basis state $|x\rangle$.

$$|x\rangle|0\rangle \rightarrow |x\rangle(F|x\rangle) \tag{5.3}$$
$$(F|x\rangle)|0\rangle \rightarrow (F|x\rangle)|x\rangle \tag{5.4}$$
$$|y\rangle|0\rangle \rightarrow |y\rangle(F^\dagger|y\rangle) \tag{5.5}$$

Note 5.3 Notice that equation (5.3) holds *only* for computational basis input states $|x\rangle$. The behaviour of QFS on superposition states is defined by linearity and it does *not* fulfil equation (5.3)! If QFS has mapped all superposition states $|x_0\rangle$ by $|x_0\rangle|0\rangle \rightarrow |x_0\rangle(F|x_0\rangle)$, then, by applying $F^\dagger$ on the second part of the state, we would be able to yield $|x_0\rangle|x_0\rangle$, i.e. copy an unknown quantum state. This would violate the no-cloning theorem.

It turns out that equation (5.3) is fulfilled for *no* superposition state. If there exists a superposition state $|\varphi\rangle$, that is mapped in such a way, then we would be able to copy exactly at least 2^n+1 distinct quantum states (all computational basis states and $|\varphi\rangle$). By repeating quantum measurements on several copies, we would be able to distinguish between them. However this is feasible only only for members of a fixed orthogonal set. Every orthogonal set on n qubits contains at most 2^n vectors.

The QFP has been defined by equation (5.4). Let us substitute $|y\rangle = F|x\rangle$, thus also $|x\rangle = F^\dagger|y\rangle$. It is tempting to define QFP by a similar equation (5.5), because this resembles equation (5.3) of the QFS more. It would be correct iff we declare $|y\rangle$ to be a *Fourier basis state*. However the mapping defined by equation (5.5) for *computational basis states* $|y\rangle$ is *not* the QFP — it follows from the previous paragraph and it can be also verified on a simple counterexample.

Theorem 5.4 [CW00] The QFT can be computed by composing the QFS and the inverse of the QFP.

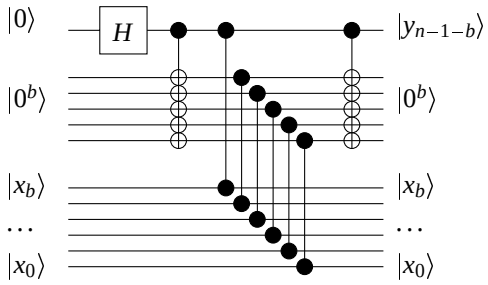

Figure 5.2: The parallelised shallow circuit computing an output qubit of the QFS

Proof. Straightforward: $|x\rangle|0\rangle \rightarrow |x\rangle(F|x\rangle) \rightarrow |0\rangle(F|x\rangle)$. Notice that the QFP is applied on registers in reverse order. □

This method requires $\Theta(n)$ ancilla qubits to store intermediate results. If we can use them, we can compute the QFT by composing the QFS and the QFP. It turns out that shallower circuits can be achieved using this decomposition.

5.4 Parallelising QFS

Let $|x\rangle$ be a computational basis state. The mapping QFS is defined by equation (5.3). Remember, that F can be decomposed using equation (5.2). Output qubits of F can be computed in parallel, since they are completely independent. It is only needed to preserve the values of source qubits, which is not a problem, because the results are stored into target qubits. From equation (5.2), each output qubit can be obtained by simply applying a sequence of rotations controlled by source qubits.

Theorem 5.5 The QFS on n qubits can be computed exactly by a quantum circuit with fan-out of constant depth and using $O(n^2)$ ancilla qubits.

Proof. In Figure 5.2, there is outlined a quantum circuit with fan-out computing the output qubit $|y_{n-1-b}\rangle = |\mu_{0.x_b \ldots x_1 x_0}\rangle = (|0\rangle + e^{2\pi i\,(0.x_b \ldots x_1 x_0)}|1\rangle)/\sqrt{2}$. It follows from the parallelisation method introduced in Chapter 4. After the Hadamard operation is applied on the target qubit, it is copied by the fan-out operation. The controlled rotations are then applied and, finally, the ancilla qubits are cleared. The ensemble of the controlled rotations yields the desired output qubit $|y_{n-1-b}\rangle$.

To obtain all output qubits of the QFS in parallel, we first need to copy source qubits using the fan-out operation (since each source qubit controls $O(n)$ rotations). We then substitute the circuit from Figure 5.2 at appropriate places (they can overlap in the first layer). The total cost is $O(n^2)$ ancilla qubits and 4 layers. The presented circuit is exact (however, only if we are able to perform the controlled rotations exactly, which can be a problem for large n). □

Corollary 5.6 $\mathrm{QFS} \in \mathrm{QNC}^0_f$. Since the fan-out n operation can be trivially simulated by a binary tree of controlled NOT operations of depth $O(\log n)$, also $\mathrm{QFS} \in \mathrm{QNC}^1$.

The approximate version of the QFS can be performed by simply omitting rotations by small phases, as it was done for the former circuit. To achieve precision ε of the circuit, the threshold $\delta = \varepsilon/\sqrt{n}$ should be chosen. The pruned circuit has depth $O(\log\log(n/\varepsilon))$ and size $O(n \cdot \log(n/\varepsilon))$.

5.5 Implementing QFP

Having implemented QFS very easily in constant depth, one would expect the same for QFP.[1] If we succeed, then also QFT could be performed in constant depth. Unfortunately QFP seems to be much more difficult than QFS. Although we have no exact proof, we can present some clues concerning the approximate computation:

There exists an approximate circuit performing the QFS with arbitrary precision ε. Each output qubit depends on only $O(\log(n/\varepsilon))$ input qubits in the circuit. If there exists a similar circuit approximating the QFP, then also the QFT could be approximated well (because the errors of consecutive operations are just summed) even if it ignores some input qubits. However it can be proved that to approximate the QFT well enough, all input qubits must be read.

Before we present this analysis, let us summarise, what is known about implementations of the QFP. It is an open problem whether there exists an exact circuit of depth $O(\log n)$ performing the QFP (in both models with / without fan-out). However, in [CW00], there has been published a quantum circuit of depth $O(\log n + \log\log(1/\varepsilon))$ and size $O(n \cdot \log(n/\varepsilon))$,

[1] The author of this thesis believed for several weeks, that the QFP is like the QFS. We mistook equation (5.5) instead of (5.4), and implemented the QFP by a similar circuit (using Lemma 5.1, it suffices to negate angles of controlled rotations). It is shown in Note 5.3 that this approach is wrong.

which performs the approximate QFP on n qubits with precision ε. The circuit is rather complicated (in comparison with circuits defined here). It is a classical quantum circuit without fan-out. Hence, for exponentially small error $\varepsilon = 2^{-O(n)}$, the QFP can be approximated in QNC^1. The size of the circuit is the smallest for polynomially small error $\varepsilon = 1/O(p(n))$.

Theorem 5.7 [CW00] The QFP on n qubits can be approximated with precision ε by a quantum circuit having depth $O(\log n + \log\log(1/\varepsilon))$ and size $O(n \cdot \log(n/\varepsilon))$.

Proof. Let us just outline the method. For details, read the article. Let us assume $|y\rangle$ is a Fourier basis state, i.e. $|y\rangle = F|x\rangle$ for a computational basis state $|x\rangle$. Again, the correctness for superposition states follows from linearity.

1. It first copies exactly the input Fourier basis state $|y\rangle$ by a so-called reversible prefix addition. This is possible in principle, since the Fourier basis is orthogonal.

   The description of the reversible prefix addition is beyond the scope of this document. For our purposes, it is only important, that it behaves like the controlled NOT for computational basis states.

2. Having constructed many copies of $|y\rangle$, it estimates statistically the computational basis output state $|x\rangle = F^\dagger|y\rangle$ and stores it into the second register. The estimation proceeds as follows.

   - The $(n-1-b)$-th qubit of $|y\rangle = F|x\rangle$ equals

     $$|y_{n-1-b}\rangle = |\mu_{0.x_b\ldots x_1x_0}\rangle = (|0\rangle + e^{2\pi i\,(0.x_b\ldots x_1x_0)}|1\rangle)/\sqrt{2},$$

     i.e. the phase of $|y_{n-1-b}\rangle$ is determined mostly by x_b, however it is perturbed by lower bits $x_{b-1}, x_{b-2}, \ldots$.

   - Let us assume the phase of $|y_{n-1-b}\rangle$ is in $\langle\frac{1}{4}\pi, \frac{3}{4}\pi\rangle \cup \langle\frac{5}{4}\pi, \frac{7}{4}\pi\rangle$. By measuring with respect to the basis $\{|\mu_{0.01}\rangle, |\mu_{0.11}\rangle\}$, it can be tested whether $x_b = 0$ or $x_b = 1$, and the correct result is obtained with probability at least $\frac{3}{4}$.

   - Let us assume the phase of $|y_{n-1-b}\rangle$ is in $\langle-\frac{1}{4}\pi, \frac{1}{4}\pi\rangle \cup \langle\frac{3}{4}\pi, \frac{5}{4}\pi\rangle$. By measuring with respect to the Hadamard basis $\{|\mu_{0.0}\rangle, |\mu_{0.1}\rangle\}$, it can be tested whether $x_b = x_{b-1}$ or $x_b = 1 - x_{b-1}$, and the correct result is obtained with probability at least $\frac{3}{4}$.

- Many measurements of either types are performed on individual copies. The obtained results **0**, **1**, **P**, and **N** denote that probably $x_b = 0$, $x_b = 1$, $x_b = x_{b-1}$, and $x_b = 1 - x_{b-1}$. The most frequent result is chosen for each b.
- The bits $x_0, x_1, \ldots, x_{n-1}$ are estimated in parallel. Notice that x_0 is always estimated exactly, i.e. only one result appears.

  Performing this step sequentially would lead to a $O(n)$ depth circuit. Using the fact, that the results can be represented by 2×2 matrices

  $$\mathbf{0} = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}, \mathbf{1} = \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}, \mathbf{P} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \mathbf{N} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

  and computing the value of x_b can be done by multiplying these matrices (which is associative), we can estimate each bit by a binary tree of depth $O(\log n)$.
- The quantum measurements are indeed not performed. Using the principle of deferred measurements ([NC00, page 186]), they are replaced by controlled quantum operations.

3. Finally, it uncomputes the reversible prefix addition, leaving only one copy of the input state $|y\rangle$.

With high probability, the phase estimation was correct, and the rest of the circuit is exact. It suffices to produce $O(\log(n/\varepsilon))$ copies to achieve precision ε. □

Note 5.4 We do not suppose, that this method can be implemented in smaller depth by a quantum circuit with fan-out, because the algorithm involves reducing a sequence of length n by an associative but not commutative operation. The fan-out operation does not help here.

By joining approximate versions of the QFS and the QFP, the following result is obtained. It is essential for several presented circuits.

Corollary 5.8 [CW00] The QFT on n qubits can be approximated with precision ε by a quantum circuit having depth $O(\log n + \log\log(1/\varepsilon))$ and size $O(n \cdot \log(n/\varepsilon))$.

Note 5.5 There might be a possible objection, why do we actually deal with the QFP. The QFS already computes the desired Fourier transform into the second register, so why we do not simply forget the value of the

input register? The reason is the quantum interference. If we forget the value of the first register, the pure quantum state would become mixed and a bulk of the entanglement would be lost.

5.6 Precision of the approximate QFT

Let us investigate, what is the error of the approximate QFT $\tilde{F}$ (according to Definition 2.3). It is distinguished from the exact QFT F by the absence of small controlled rotations — for each target qubit, not more than k rotations of phase $\Omega\left(2^{-k}\right)$ are performed on it. The trace distance is defined in Definition 2.2.

Lemma 5.9 Let $\tilde{F}$ be the approximate QFT that performs only rotations of phase at least $\delta = \pi \cdot 2^{-k}$. Then, for each computational basis state $|x\rangle$, the trace distance between $\tilde{F}|x\rangle$ and $F|x\rangle$ is $\mathrm{O}(\sqrt{n} \cdot \delta)$.

Proof. Let $|x\rangle$ be a computational basis state. Remind, that $|\mu_{0.x_k \ldots x_1 x_0}\rangle = (|0\rangle + e^{2\pi i\,(0.x_k \ldots x_1 x_0)}|1\rangle)/\sqrt{2}$. Then

$$
\begin{aligned}
|y\rangle = F|x\rangle &= \bigotimes_{b=0}^{n-1} |\mu_{0.x_b x_{b-1} \ldots x_1 x_0}\rangle, \\
|\tilde{y}\rangle = \tilde{F}|x\rangle &= \bigotimes_{b=0}^{k} |\mu_{0.x_b x_{b-1} \ldots x_1 x_0}\rangle \otimes \bigotimes_{b=k+1}^{n-1} |\mu_{0.x_b x_{b-1} \ldots x_{b-k}}\rangle, \\
\langle \tilde{y}|y\rangle &= \prod_{b=0}^{k} \underbrace{\langle \mu_{0.x_b \ldots x_0}|\mu_{0.x_b \ldots x_0}\rangle}_{1} \prod_{b=k+1}^{n-1} \underbrace{\langle \mu_{0.x_b \ldots x_{b-k}}|\mu_{0.x_b \ldots x_0}\rangle}_{\text{rotationally invariant}} \\
&= \prod_{b=k+1}^{n-1} \langle \mu_{0.0^{k+1}}|\mu_{0.0^{k+1} x_{b-k-1} \ldots x_1 x_0}\rangle \\
|\langle \tilde{y}|y\rangle| &\geq \prod_{b=0}^{n-1} |\langle \mu_{0.0}|\mu_{0.0^k 1}\rangle| = \left|\frac{1}{2}(\langle 0| + \langle 1|)(|0\rangle + e^{2\pi i/2^{k+1}}|1\rangle)\right|^n \\
&= \left|\frac{1 + e^{\pi i/2^k}}{2}\right|^n = \left|\frac{(1+\cos\delta) + i \cdot \sin\delta}{2}\right|^n \\
&\quad \left[\begin{array}{ll} \cos 2\alpha = 2\cos^2\alpha - 1, & \sin 2\alpha = 2\sin\alpha \cdot \cos\alpha \\ \frac{1+\cos\delta}{2} = \cos^2\frac{\delta}{2}, & \frac{\sin\delta}{2} = \sin\frac{\delta}{2} \cdot \cos\frac{\delta}{2} \end{array}\right] \\
&= \left|\cos^2\frac{\delta}{2} + i \cdot \sin\frac{\delta}{2} \cdot \cos\frac{\delta}{2}\right|^n = \cos^n\frac{\delta}{2} \cdot |e^{n\delta i/2}| = \cos^n\frac{\delta}{2}.
\end{aligned}
$$

We approximate the trigonometric functions on the interval $(0,1)$ and compute the trace distance:

$$\begin{aligned}
\cos\delta &= 1-\frac{\delta^2}{2!}+\frac{\delta^4}{4!}-\ldots \geq 1-\frac{\delta^2}{2!}, \\
\mathbf{F}(\tilde{y},y)=|\langle\tilde{y}|y\rangle| &\geq \cos^n\frac{\delta}{2} \geq \left(1-\frac{\delta^2}{8}\right)^n = \sqrt{1-\frac{\delta^2}{8}}^{2n} \geq \sqrt{1-\frac{n\delta^2}{4}} \\
\mathbf{D}(\tilde{y},y)=\sqrt{1-\mathbf{F}(\tilde{y},y)^2} &\leq \sqrt{\frac{n\delta^2}{4}} = \frac{1}{2}\sqrt{n}\cdot\delta = \mathrm{O}\left(\sqrt{n}\cdot\delta\right).
\end{aligned}$$

□

Corollary 5.10 It suffices to perform $\mathrm{O}(\log(n/\varepsilon))$ controlled rotations on each qubit to achieve expected precision ε (for a random input state) of the approximate QFT.

Proof. Since the trace distance metric is equivalent with the Euclidean metric, the error of the approximation is $\varepsilon \equiv \mathbf{D}(\tilde{y},y)$. From Lemma 5.9, the threshold of rotations performed must be $\delta = \mathrm{O}(\varepsilon/\sqrt{n})$ to achieve the precision ε for each computational basis input state. The number of controlled rotations is $k = \mathrm{O}(\log(1/\delta)) = \mathrm{O}(\log(\sqrt{n}/\varepsilon)) = \mathrm{O}(\log(n/\varepsilon))$.

The error estimation was inferred for computational basis input states. From Theorem 2.1, it follows that the expected approximation error is small enough also for superposition input states.

However, to have the error of the approximate QFT guaranteed below ε for every input state $|x\rangle$, we need to achieve precision $2^{-n/2}\varepsilon$ for each computational state. Hence we need to perform

$$\Omega\left(\log\frac{n}{2^{-n/2}\varepsilon}\right) = \Omega\left(\log(n/\varepsilon)+\log 2^{n/2}\right) = \Omega(\log(1/\varepsilon)+n)$$

controlled rotations. □

The following theorem shows, that every good approximate circuit must incorporate each input qubit.

Theorem 5.11 [CW00] Let C be a circuit computing the *high-order output qubit* $|y_{n-1}\rangle$ of the QFT operation. If there exists an input qubit $|x_k\rangle$ that $|y_{n-1}\rangle$ does not depend on, then the error of C exceeds $1/10$. That is there exists an input state $|\psi\rangle$, for which the error of the approximation (Euclidean distance of the exact and approximated output) exceeds $1/10$.

Note 5.6 Let us first clarify a possible objection: Theorem 5.2 implies that $|y_{n-1}\rangle = |\mu_{0.x_0}\rangle = H|x_0\rangle$ for each computational basis input state $|x\rangle$. This suggest that it suffices to just perform the Hadamard operation on $|x_0\rangle$ and return the result. Nevertheless this does *not* work for superposition states, because the evolution $|x_0\rangle \to |y_{n-1}\rangle$ is not a closed quantum system and the state $|x_0\rangle$ interacts also with other qubits (and thus passes some information about its value to them).

In other words, for a general superposition input state $|x\rangle$, the output qubit $|y_{n-1}\rangle$ will not be in tensor product with other qubits, otherwise it would be possible to yield information about the quantum state $|x_0\rangle$ without disturbing it. The information about its value has been passed to another part of the system and that part is in tensor product with $|y_{n-1}\rangle$, which contains complete information about $|x_0\rangle$. This is a contradiction with the no-cloning theorem.

Proof of Theorem 5.11. For a computational basis state $|y\rangle = |y_{n-1}\dots y_1 y_0\rangle$, let $|\psi_y\rangle = \bigotimes_{b=0}^{n-1} |\mu_{(-0.y_b\dots y_1 y_0)}\rangle$ be the *inverse Fourier basis state* (with phase parameter y). Since $|\psi_y\rangle = F^\dagger|y\rangle$, also $F|\psi_y\rangle = |y\rangle$. Hence when C is applied on $|x\rangle = |\psi_y\rangle$, a good approximation of $|y_{n-1}\rangle$ should be yielded. (The information about $|y_{n-1}\rangle$ is stored in the input qubit $|x_0\rangle$ and it is perturbed by other qubits there: $|x_0\rangle = |\mu_{(-0.y_{n-1}\dots y_1 y_0)}\rangle$.)

Let us prove by contradiction, that if $|y_{n-1}\rangle$ does not depend on an input qubit $|x_k\rangle$ (for any $k \in \mathbf{Z}_n$), then the error exceeds $1/10$. Let $r = n-k-1$.

We first consider states $|\psi_z\rangle$, $|\psi_{z+2^r}\rangle$ for $z = 2^n - 1$. The circuit C approximates $|y_{n-1}\rangle$ with error $\leq 1/10$ on every input. Since $z = 1^n$ and $z + 2^r \equiv 0^{n-r}1^r \pmod{2^n}$ in binary, the output qubit $|y_{n-1}\rangle$ approximates $|1\rangle$ on input $|\psi_z\rangle$ and $|0\rangle$ on input $|\psi_{z+2^r}\rangle$. Recall that the trace distance is upper bounded by the Euclidean distance, hence $\mathbf{D}\left(|1\rangle, |y_{n-1}\rangle_{|x\rangle=|\psi_z\rangle}\right) \leq 1/10$ and vice versa.

Now, we consider a third state $|\psi'_z\rangle$. It resembles $|\psi_z\rangle$, but the k-th qubit (the one that is ignored) is distinct:

$$|\psi'_z\rangle = |\mu_{0.1}\rangle \dots |\mu_{0.1^r}\rangle \otimes |\mu_{0.01^r}\rangle \otimes |\mu_{0.1^{r+2}}\rangle \dots |\mu_{0.1^n}\rangle.$$

Notice that the distinct qubits are orthogonal: $\langle \mu_{0.01^r} | \mu_{0.1^{r+1}} \rangle = \langle \mu_{0.0} | \mu_{0.1} \rangle = 0$. Since $|y_{n-1}\rangle$ does not depend on $|x_k\rangle$, the circuit C must also yield an approximation of $|1\rangle$ on input $|\psi'_z\rangle$ (likewise on input $|\psi_z\rangle$).

On the other hand, the trace distance between $|\psi'_z\rangle$ and $|\psi_{z+2^r}\rangle$ can be calculated to be below 0.7712, as follows. The two states are identical in qubit positions $n-1, n-2, \dots, k$. In qubit position $k-b$ for $b \in \{1, 2, \dots, k\}$, the two states differ by an angle of $\pi/2^{b+1}$. (Remind the trigonometric

equalities used in the proof of Lemma 5.9.) In particular,

$$\begin{aligned}
\langle \psi'_z | \psi_{z+2^r} \rangle &= \prod_{b=1}^{k} \langle \mu_{0.1^{r+b+1}} | \mu_{0.0^{b+1}1^r} \rangle = \prod_{b=1}^{k} \langle \mu_{0.1^{b+1}} | \mu_{0.0^{b+1}} \rangle \\
&= \prod_{b=1}^{k} \langle \mu_{0.0} | \mu_{0.0^b 1} \rangle = \prod_{b=1}^{k} \left(\frac{1}{2} (\langle 0| + \langle 1|)(|0\rangle + e^{2\pi i/2^{b+1}} |1\rangle) \right) \\
&= \prod_{b=1}^{k} \frac{1 + e^{\pi i/2^b}}{2} = \prod_{b=1}^{k} \left(\cos \frac{\pi}{2^{b+1}} \cdot e^{\pi i/2^{b+1}} \right), \\
|\langle \psi'_z | \psi_{z+2^r} \rangle| &= \prod_{b=1}^{k} \cos \frac{\pi}{2^{b+1}} > \prod_{b=0}^{\infty} \cos \frac{\pi}{4 \cdot 2^b} > 0.6366.
\end{aligned}$$

This implies that the trace distance between $|\psi'_z\rangle$ and $|\psi_{z+2^r}\rangle$ is less than $\sqrt{1-0.6366^2} = 0.7712$. Since the trace distance is contractive, it follows that the trace distance of the states of the high-order output $|y_{n-1}\rangle$ of C on inputs $|\psi'_z\rangle$ and $|\psi_{z+2^r}\rangle$ is also less than 0.7712. But, by the triangle inequality

$$\begin{aligned}
\mathbf{D}(|0\rangle, |1\rangle) &\leq \mathbf{D}\left(|0\rangle, |y_{n-1}\rangle_{|x\rangle=|\psi_{z+2^r}\rangle}\right) + \mathbf{D}\left(|y_{n-1}\rangle_{|x\rangle=|\psi_{z+2^r}\rangle}, |y_{n-1}\rangle_{|x\rangle=|\psi_z\rangle}\right) + \\
&\quad + \mathbf{D}\left(|y_{n-1}\rangle_{|x\rangle=|\psi_z\rangle}, |1\rangle\right) \leq \\
&\leq \frac{1}{10} + \mathbf{D}(|\psi_{z+2^r}\rangle, |\psi_z\rangle) + \frac{1}{10} \leq \frac{1}{10} + 0.7712 + \frac{1}{10} = 0.9712 < \\
&< 1,
\end{aligned}$$

which is a contradiction, since $|0\rangle$ and $|1\rangle$ are orthogonal. □

Note 5.7 The state $|\psi'_z\rangle$ needs to be used, since the scalar product of the original vectors $\langle \psi_z | \psi_{z+2^r} \rangle = 0$ due to involving $\cos \frac{\pi}{2} = 0$ in the product.

Conclusion 5.12 [CW00] There exist small (narrow and shallow) quantum circuits approximating the QFT with exponentially small error. To achieve precision better than $1/10$ for all input states, every output qubit needs to depend on all lower-order input qubits.

Chapter 6

Increment operation

Definition 6.1 An *increment operator* (denoted by P) on n qubits is an operator mapping each computational basis state $|x\rangle$ to $|x+1 \bmod 2^n\rangle$.

6.1 Classical circuit

Let us first review an efficient classical circuit for the increment operator. We then convert it to a quantum circuit using standard methods.

Lemma 6.1 n-bit numbers can be incremented by a classical (irreversible) circuit of depth $O(\log n)$ using $O(n)$ ancilla bits.

Proof. The input n-bit number is denoted by $x = x_{n-1} \ldots x_1 x_0$. The output number $y = x+1$ is denoted by $y = y_{n-1} \ldots y_1 y_0$. It is easy to sea, that $y_k = x_k \oplus 1$ iff all lower-order input bits $x_0, x_1, \ldots, x_{k-1}$ are set, otherwise $y_k = x_k$. In other word, $y_k = x_k \oplus \bigwedge_{j=0}^{k-1} x_j$. Let $c_k = \bigwedge_{j=0}^{k} x_j$ denote the k-th carry bit. If we have computed all carry bits c_k, we can compute y in one layer by setting $y_k = x_k \oplus c_{k-1}$.

We have reduced incrementing a number to computing all carry bits, i.e. computing logical conjunctions of all prefixes of input variables $c_k = \bigwedge_{j=0}^{k} x_k$. From associativity of $\wedge$, computing a conjunction of one such prefix can be done in depth $O(\log n)$ by a balanced tree of AND gates. To compute all conjunctions in parallel, we need to reuse temporary results.

Let us assume w.l.o.g. that $n = 2^m$, otherwise we cut a left part of the circuit. In Figure 6.1, there is outlined a classical circuit that, for inputs $\{x_k\}_{k=0}^{n-1}$, yields outputs $\{c_k\}_{k=0}^{n-1}$. Every arrow going from x_a to x_b denotes setting $x_b := x_b \wedge x_a$. The right dashed line represents a constant 1 and it is included just for symmetry of the figure (and it can be omitted with all adjacent arrows).

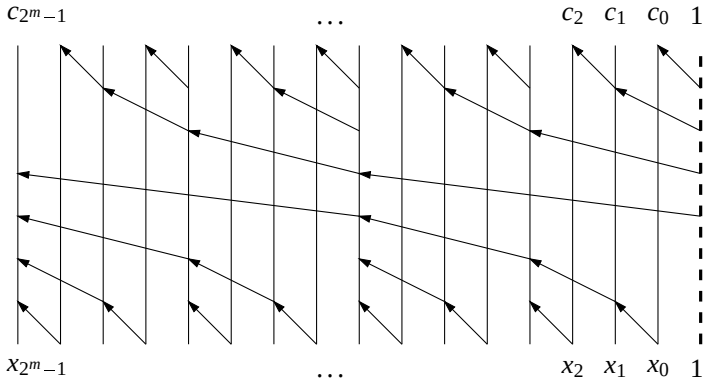

Figure 6.1: Computing logical conjunctions of all prefixes $c_k = \bigwedge_{j=0}^{k} x_k$

After the first part of the circuit, each bit contains the conjunction of input variables from its corresponding subtree. In the second part, carry bits are computed by merging adjacent blocks. The correctness of the circuit is straightforward.

The total circuit uses $O(n)$ ancilla bits for storing the carry bits and it has depth $O(\log n)$. □

Theorem 6.2 *n*-bit numbers can be incremented by a classical reversible circuit of depth $O(\log n)$ using $O(n)$ ancilla bits.

Proof. We convert the classical irreversible circuit from Lemma 6.1 into a reversible one. Since there are $O(n)$ irreversible AND operations in total, only $O(n)$ new ancilla bits are needed (instead of replacing a bit, we allocate a new one).

Having computed all carry bits, we compute the output number y and store it to target bits. We then uncompute the carry bits. The depth is doubled, but is stays $O(\log n)$. □

Corollary 6.3 There exists a quantum circuit computing the increment operation exactly in depth $O(\log n)$ using $O(n)$ ancilla qubits. Hence the increment operator $P \in \mathrm{QNC}^1$.

6.2 Quantum circuit

Let us investigate the diagonal form of the increment operator P. A simple argument yields that P is diagonal in the Fourier basis. Let F be the QFT.

Theorem 6.4 $P = F^\dagger DF$, where $D = \mathrm{Diag}(1, \xi, \xi^2, \ldots, \xi^{2^n-1}) = \sum_{x=0}^{2^n-1} \xi^x |x\rangle\langle x|$ is an operator diagonal in the computational basis and $\xi = e^{2\pi i/2^n}$.

Proof. Let $|x\rangle$ be a computational basis state. Using equation (5.1) on page 29,

$$\begin{aligned} F^\dagger DF|x\rangle &= F^\dagger D \sum_{y=0}^{2^n-1} \xi^{yx}|y\rangle = F^\dagger \sum_{y=0}^{2^n-1} \xi^y \cdot \xi^{yx}|y\rangle = F^\dagger \sum_{y=0}^{2^n-1} \xi^{y(x+1)}|y\rangle \\ &= F^\dagger \sum_{y=0}^{2^n-1} \xi^{y(x+1 \bmod 2^n)}|y\rangle = F^\dagger(F|x+1 \bmod 2^n\rangle) \\ &= |x+1 \bmod 2^n\rangle = P|x\rangle. \end{aligned}$$

The equality for superposition states follows from linearity. □

Next, let us show that the operator D is easy to implement. It turns out that D operates on each qubit separately, i.e. it is a tensor product of one qubit operators. Let us fix n and set $D = D_n$, where $D_k = \mathrm{Diag}(1, \xi, \ldots, \xi^{2^k-1})$ is a k-qubit operator (diagonal in the computational basis).

Remind, that the rotation operator rotation about the z-axis by angle θ is defined by $R_z(\theta) = |0\rangle\langle 0| + e^{\theta i}|1\rangle\langle 1|$.

Theorem 6.5 For every $k \in \{1, 2, \ldots, n\}$, $D_k = R_z(\pi/2^{n-k}) \otimes D_{k-1}$. The trivial 0-qubit operator D_0 is considered to be $D_0 = 1$.

Proof. A tensor product of two diagonal operators is a diagonal operator. Hence we need to verify only the equality of diagonal elements. Let $|x\rangle = |x_{k-1}\rangle|\overline{x}\rangle$ be a k-qubit computational basis state ($\overline{x} = x_{k-2} \ldots x_1 x_0$). From the definition of D_k, $\langle x|D_k|x\rangle = \xi^x$ and $\langle \overline{x}|D_{k-1}|\overline{x}\rangle = \xi^{\overline{x}}$. On the other hand,

$$\begin{aligned} \langle x|R_z(\pi/2^{n-k}) \otimes D_{k-1}|x\rangle &= \langle x_{k-1}|\langle \overline{x}| \left(R_z(\pi/2^{n-k}) \otimes D_{k-1}\right) |x_{k-1}\rangle|\overline{x}\rangle \\ &= \langle x_{k-1}|R_z(\pi/2^{n-k})|x_{k-1}\rangle \cdot \langle \overline{x}|D_{k-1}|\overline{x}\rangle \\ &= e^{\pi/2^{n-k} \cdot i \cdot x_{k-1}} \cdot \xi^{\overline{x}} = e^{(2\pi i/2^n)\cdot(2^{k-1}x_{k-1})} \cdot \xi^{\overline{x}} \\ &= \xi^{(2^{k-1}x_{k-1})} \cdot \xi^{\overline{x}} = \xi^{(2^{k-1}x_{k-1}+\overline{x})} = \xi^x, \end{aligned}$$

which is the desired value. The case $D_0 = 1$ is obvious. □

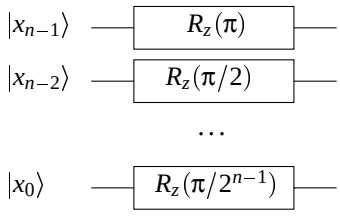

Figure 6.2: Diagonal form $D = FPF^\dagger$ of the increment operator P

Corollary 6.6 The increment operator P is diagonal in the Fourier basis and, in this basis, it can be implemented exactly by a constant depth quantum circuit. It consists of just one layer.

Hence the operator $D = FPF^\dagger$ (the increment in the Fourier basis, i.e. the operator mapping $F|x\rangle \to F|x+1 \bmod 2^n\rangle$) is in QNC^0.

Proof. Using Theorem 6.5 and mathematical induction, the operator D can be implemented by a simple depth 1 circuit shown in Figure 6.2. □

Note 6.1 If we need to work in the computational basis, i.e. implement the operator P, we have to precede D by F and succeed it by $F^\dagger$. The QFT can be approximated with exponential precision by a $\mathrm{O}(\log n)$-depth quantum circuit. Hence the increment operation can be approximated in QNC^1.

This result is not interesting, as is, because the classical circuit has also depth $\mathrm{O}(\log n)$ and it is exact. However the strength of this method is the capability of being parallelised by the method presented in Chapter 4, since D is diagonal in the computational basis. This result is original.

Note 6.2 Since $P^k = F^\dagger D^k F$ and $R_z(\alpha)^k = R_z(k\alpha)$, the operator P^k adding a fixed number k can be also implemented by the presented method. One just needs to multiply the rotations of the operator D by k.

Chapter 7

Interesting shallow circuits

In this chapter, we implement quantum circuits with fan-out for Toffoli, Majority, Threshold, and Counting gates having sub-logarithmic depth. More methods with distinct depths, space complexities, and precisions are presented. Results presented in this chapter are original.

Using a similar method, we also mention how to implement approximations of two basic arithmetical operators (summing and multiplying) in logarithmic depth.

Definition 7.1 A *linear Threshold* gate G on n source qubits $|x_1\rangle, |x_2\rangle, \ldots, |x_n\rangle$ and one target qubit $|t\rangle$ with *weights* $\{\alpha_k\}_{k=1}^n$ and *threshold* m is a quantum operation performing (written in the computational basis)

$$\left(\bigotimes_{k=1}^{n} |x_k\rangle\right) |t\rangle \to \left(\bigotimes_{k=1}^{n} |x_k\rangle\right) |t \oplus \operatorname{sgn}(z)\rangle, \qquad z = \sum_{k=1}^{n} \alpha_k x_k - m \tag{7.1}$$

for each computational basis state and the behaviour for superposition states is defined by linearity. Recall that $\operatorname{sgn}(z) = 1$ iff $z \geq 0$, hence the gate flips the target qubit if the linear combination of source qubits exceeds the threshold. We assume the gate parameters $\alpha_k, m \in \mathbf{Z}$.

A *uniform Threshold* gate is a linear Threshold gate with weights $\alpha_k = 1$ for each k. A *Majority* gate is a uniform Threshold gate with threshold $m = \lceil n/2 \rceil$. A *Toffoli* (AND) gate is a uniform Threshold gate with threshold $m = n$.

Definition 7.2 TC^k is a class of problems solvable by circuits of depth $\mathrm{O}\left(\log^k n\right)$. It is allowed to use the same gates as in AC^k (i.e. unbounded fan-in AND and OR gates, and the NOT gate) and the linear Threshold gate. QTC^k is its quantum variant (inferred from QAC^k by adding the linear Threshold gate).

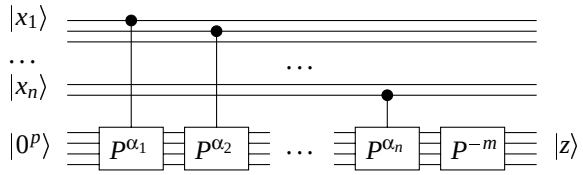

Figure 7.1: A serial circuit implementing the linear Counting gate

Definition 7.3 A *linear Rank* gate behaves similarly to the linear Threshold gate from Definition 7.1, but the target qubit t is flipped iff $z = 0$, i.e. the *equality* $\sum_{k=1}^{n} \alpha_k x_k = m$ is tested. A *uniform Rank* gate is a linear Rank gate with weights $\alpha_k = 1$ for each k.

Note 7.1 The Toffoli gate is also a special case of the uniform Rank gate with rank $m = n$.

Definition 7.4 A *linear Counting* gate on n qubits is a quantum operation that, for each computational basis input state, computes the integer number $z = \sum_{k=1}^{n} \alpha_k x_k - m$. (It assumes ancilla qubits are cleared on the input and it stores the result there.) On superposition states, it is defined by linearity. A *(uniform) Counting* gate is a linear Counting gate with weights $\alpha_k = 1$ for each k, i.e. it outputs the number of source qubits set to 1.

7.1 Approximative Counting and Threshold gate

Let us develop a shallow circuit for the linear Counting gate by incorporating presented techniques. The desired value z is computed and stored in numerical register Z. We have already implemented an efficient and easily diagonalisable version of the operator P^a, adding an arbitrary fixed integer number a to Z. Hence it suffices if, for each source qubit x_k, we apply the operator P^{α_k} controlled by x_k. After this, we subtract m by applying P^{-m}. Since the operators P^{α_k} commute, they can be parallelised by the parallelisation method.

The size p of the register Z needs to be chosen big enough to carry any number that can arise during the computation. Let g, h be the minimal and maximal possible value of z. It suffices to have Z consisted of $p = 1 + \lceil \log_2(1 + \max(|g|, |h|)) \rceil$ qubits. (The $+1$ qubit is for the sign of the result.) If all weights are 1, then $p = \mathrm{O}(\log n)$. For general weights bounded by f, $p = \mathrm{O}(\log nf) = \mathrm{O}(\log n + \log f)$.

A raw serial circuit implementing the desired operation is outlined in Figure 7.1. This circuit needs to be parallelised. We already know that $P^a = F^\dagger D^a F$, where F is the Quantum Fourier Transform, and D is a diagonal operator that can be represented by a one-layer circuit consisting of one qubit gates.

Theorem 7.1 The linear Counting gate operating on n qubits with polynomially bounded parameters (and thus also the simpler uniform Counting gate) can be approximated with polynomially small error by a quantum circuit with fan-out of depth $\mathrm{O}(\log\log n)$ operating on $\mathrm{O}(n\log n)$ ancilla qubits.

Proof. We apply the simplified[1] parallelisation method (described by Theorem 4.1) to the serial circuit shown in Figure 7.1: The ancilla qubits are converted into the Fourier basis by applying F in the beginning and $F^\dagger$ at the end of the raw circuit. Furthermore, each operator P^a is replaced by its diagonal form D^a. Let $f = \mathrm{O}(p_1(n))$ be the number bounding the gate parameters (all weights α_k and the constant number m). The ancilla register Z consists of $p = \mathrm{O}(\log n + \log f) = \mathrm{O}(\log n)$ ancilla qubits. Let $\varepsilon = 1/\mathrm{O}(p_2(n))$ be the approximation error.

The depth: Using Corollary 5.8, the QFT F can be approximated in $\mathrm{O}(\log p + \log\log(1/\varepsilon)) = \mathrm{O}(\log\log n + \log\log p_2(n)) = \mathrm{O}(\log\log n)$ layers and we need to perform it twice. Furthermore, 2 layers are spent in copying the ancilla qubits using the fan-out operation (and clearing them back). Each operator D^a can be implemented in 1 layer, however a controlled version needs to be performed. Using Theorem 3.3, this can be done in 5 layers. The total depth is double-logarithmic.

Ancilla qubits: The register Z has size $p = \mathrm{O}(\log n)$. The parallelisation method uses $\mathrm{O}(n\log n)$ ancilla qubits (to store $n+1$ copies of Z). The QFT operator uses $\mathrm{O}(p\log(p/\varepsilon)) = \mathrm{O}(\log n \cdot \log(\log n \cdot p_2(n)) = \mathrm{O}\left(\log^2 n\right)$ ancilla qubits. The total space complexity is $\mathrm{O}(n\log n)$. □

Theorem 7.2 The linear Threshold gate operating on n qubits with polynomially bounded parameters (and thus also the simpler uniform Threshold, Majority, and Toffoli gates) can be approximated with polynomially small error by a quantum circuit with fan-out of depth $\mathrm{O}(\log\log n)$ operating on $\mathrm{O}(n\log n)$ ancilla qubits.

[1] It is simplified in the sense that we already have an efficient implementation of the diagonal form of the increment operator and hence we need not perform the QFT four times, but only twice.

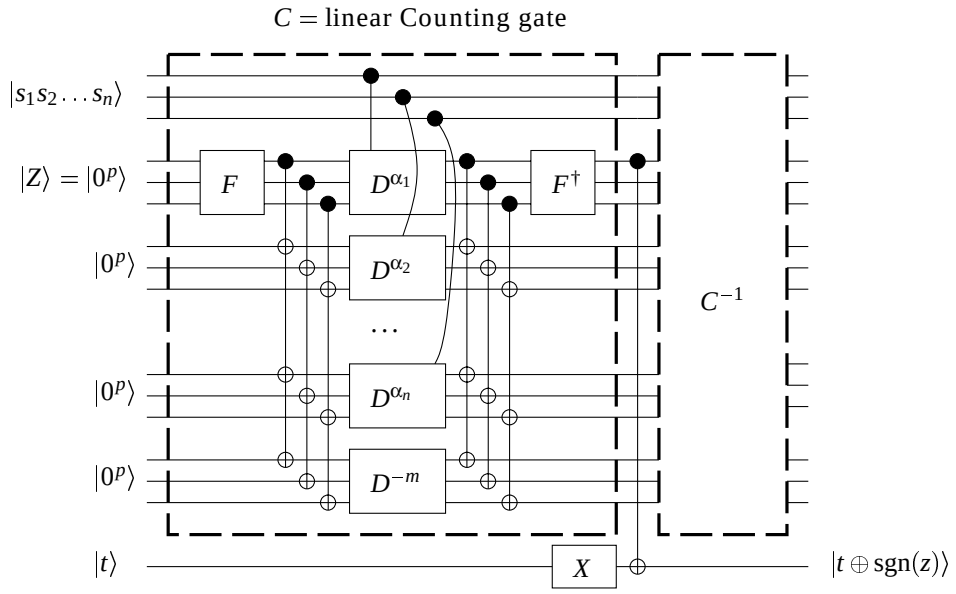

Figure 7.2: A parallel circuit implementing the linear Threshold gate

Proof. Using the linear Counting gate described by Theorem 7.1, we compute the value of z. The sign of z can be obtained by simply negating the most significant qubit of the register Z, because Z is just a simple signed integer register. We negate the target qubit t iff $\mathrm{sgn}(z) = 1$ and clear all ancilla qubits by uncomputing z. The total depth stays double-logarithmic and also the space complexity is preserved. The final circuit (after the parallelisation method has been performed) is outlined in Figure 7.2. □

7.2 Approximative Rank and Toffoli gate

Having implemented the Counting gate, we can also readily obtain the Rank gate — it suffices, instead of looking at the sign of z, to compare z with 0, e.g. by applying the uniform Threshold gate on z. It immediately follows that the Rank gate can be approximated well in depth $\mathrm{O}(\log\log n)$. Nevertheless, a better depth can be achieved if we incorporate the Chinese Remainder Theorem.

Theorem 7.3 The Rank gate (and hence also the Toffoli gate) can be approximated with precision ε by a quantum circuit with fan-out having

depth $O((\log\log\log n)\log(\log\log n+\log(1/\varepsilon)))$ and using $O(n\log n\log\log n)$ ancilla qubits.

Proof. Let us just outline the method, because it will be improved by a forthcoming method that is exact and has smaller depth. Let us assume w.l.o.g., that all weights are positive integers.

We choose a set of integer numbers $M=\{m_k\}$ such, that $m=\prod_k m_k$ is bigger than the biggest possible value v of Z and $\gcd(m_k,m_l)=1$ for each $k\neq l$. From Chinese Remainder Theorem, $z_1\equiv z_2 (\mathrm{mod}\, m)$ iff $z_1\equiv z_2(\mathrm{mod}\, m_k)$ for each k. It can be shown, that M can be chosen in such a way, that $\#M=O(\log v)$ and $m_k=O(\log v)$ for each k.

We will not compute the actual value of z, but remainders $z \mathrm{mod} m_k$ for each k. To do that, we need to replace the regular QFT (modulo a power of 2) by the generalised QFT (modulo an arbitrary integer) and perform the increment operation modulo the same integer. From [CW00], the QFT modulo q can be approximated with error δ by a quantum circuit of depth $O((\log\log q)(\log\log(1/\delta)))$. There exists a classical circuit for the operation increment modulo q of depth $O(\log\log q)$.

After we have computed all $\#M=O(\log n)$ remainders in parallel, we compare each of them with the expected value and perform the Toffoli (AND) gate on the results. It suffices to use the simpler method working in depth $O(\log\log\#M)=O(\log\log\log n)$ with polynomially small error.

Since $v=O(p(n))$, each $m_k=O(\log v)=O(\log n)$. Hence the computation of $z \mathrm{mod} m_k$ can be done in depth $O((\log\log\log n)(\log\log(1/\delta)))$ using $O(n\log m_k)=O(n\log\log n)$ ancilla qubits. Since we perform and combine $O(\log\#M)=O(\log n)$ independent computations, the total error is $\varepsilon=O(\delta\log n)$.

To achieve precision ε, the constructed quantum circuit will be of depth $O((\log\log\log n)(\log\log(\log n/\varepsilon)))$ and it will use $O(n\log n\log\log n)$ ancilla qubits. □

Corollary 7.4 The Rank gate can be approximated with logarithmically small error by a quantum circuit with fan-out of depth $O\left(\log^2\log\log n\right)$.

7.3 Exact Rank and Toffoli gate

To achieve circuits shallower than $O(\log\log n)$, we need to get rid of the QFT, because it seems to have at least logarithmic depth. It turns out, that if we replace the QFT by the Hadamard transform (which has constant depth) and preserve the rest of the circuit, the modified circuit keeps

working in a special case. We no longer obtain the exact number z of source qubits set in the register Z, but the final quantum state contains enough information to identify an arbitrary fixed value with probability 1.

Definition 7.5 The *Hadamard transform* on n qubits is the following operator (written in the computational basis):

$$H_n = \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} |y\rangle \sum_{x=0}^{2^n-1} (-1)^{y \bullet x} \langle x|,$$

where $y \bullet x$ is the bitwise scalar product. The Hadamard operator on 1 qubit is also denoted by $H = H_1$.

Note 7.2 The Hadamard operator is essential for quantum computation and it is defined in every textbook of Quantum Computing, e.g. in [NC00]. It has a very simple and well known decomposition, which leads to an efficient depth 1 quantum circuit.

Theorem 7.5 $H_n = H^{\otimes n}$.

Proof. Let x,y be any computational basis states. Let $|x\rangle = |x_{n-1} \ldots x_1 x_0\rangle$ and $|y\rangle = |y_{n-1} \ldots y_1 y_0\rangle$. Then

$$\langle y|H_n|x\rangle = 2^{-n/2}(-1)^{y \bullet x} = \prod_{k=0}^{n-1} \frac{(-1)^{y_k x_k}}{\sqrt{2}} = \prod_{k=0}^{n-1} \langle y_k|H|x_k\rangle = \langle y|H^{\otimes n}|x\rangle,$$

which implies the operators are equal. □

Theorem 7.6 The linear Rank gate operating on n qubits with polynomially bounded parameters can be computed exactly by a quantum circuit with fan-out of depth $O(\log^* n)$ operating on $O(n \log n)$ qubits.

Proof. We use the same method as in Theorem 7.2, but we replace the QFT by the Hadamard transform. Let us compute, what happens to the register Z during the computation. Since Z is cleared in the beginning of the computation, it does not matter whether the QFT or the Hadamard transform is used — anyway it is mapped to the state

$$Z = F|0\rangle = \frac{1}{2^{p/2}} \sum_{y=0}^{2^p-1} |y\rangle = H_p|0\rangle = H^{\otimes p}|0\rangle.$$

Let the input qubits be in a computational basis state. (If the input qubits are in a superposition, the correctness of the method follows from linearity.) The total phase shift (number of increments in the Fourier basis)

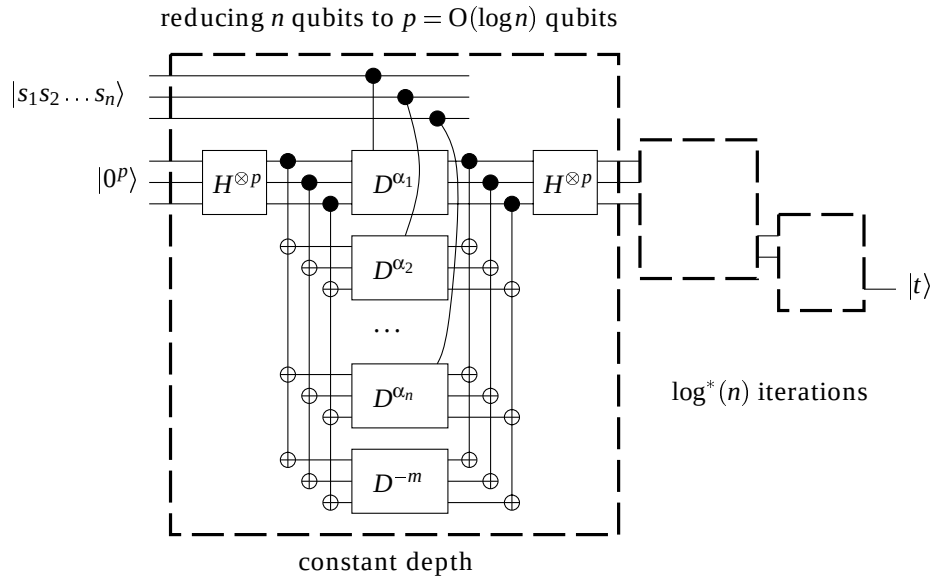

Figure 7.3: The Rank gate consists of an iterated $n \to O(\log n)$ reduction

gained by Z is denoted by z. At the end of the computation, we need to check, whether $z = 0$. Let $\xi = e^{2\pi i/2^p}$. The register Z is mapped by the second Hadamard transform to the final state

$$H\left(\frac{1}{2^{p/2}}\sum_{y=0}^{2^p-1}\xi^{zy}|y\rangle\right) = \frac{1}{2^p}\sum_{x=0}^{2^p-1}|x\rangle\sum_{y=0}^{2^p-1}(-1)^{x\bullet y}\cdot\xi^{zy}.$$

The amplitude of the zero state $|0\rangle$ is $2^{-p}\cdot\sum_{y=0}^{2^p-1}\xi^{zy}$. If $z = 0$, then this amplitude is 1, otherwise it is 0 (see the proof of Lemma 5.1 for details of computing the sum).

Let the input qubits be in computational basis state $|x_1 x_2 \ldots x_n\rangle$. If they fulfil the original equation $z = \sum_{k=1}^{n}\alpha_k x_k - m = 0$, then the register $Z = Z_{p-1}\ldots Z_1 Z_0$ is in computational basis state $|0\rangle$ with amplitude 1, i.e. its bits fulfil a new equation $\sum_{k=0}^{p-1} Z_k = 0$. Otherwise the register Z is in a general superposition state, and we know that *no* computational basis state in this superposition fulfils the new equation. That is the new equation is *equivalent* to the original equation.

Hence we have reduced the question of whether n input variables fulfil a linear equation, to the question of whether $p = O(\log n)$ ancilla qubits fulfil another equation of the same type. We can thus iterate the method. After $\log^*(n)$ iterations, the number of ancilla qubits is bounded by a con-

stant and we can use a normal exact method of depth $O(1)$. We then copy the final result into another ancilla qubit and uncompute everything. The circuit is outlined in Figure 7.3, one iteration corresponds to a dashed box. (For clarity, unimportant ancilla qubits arise and end in the middle of the computation and the uncomputation is omitted.)

Every iteration except for the last one can be performed by a constant depth circuit consisting of the following 9 layers: 2 for the Hadamard transform, 2 for the fan-out operation, and 5 for the controlled rotations. The last iteration also takes $O(1)$ layers. Everything is doubled since we need to uncompute intermediate results. The total depth is $O(\log^* n)$. The number of ancilla qubits is $O(n \log n)$, they all are spent in the first iteration, the other ones need less ancilla qubits. □

Corollary 7.7 The Toffoli gate on n qubits can be computed exactly by a quantum circuit with fan-out of depth $O(\log^* n)$ operating on $O(n \log n)$ ancilla qubits.

Note 7.3 It is quite interesting, that even for computational basis input states, the registers $Z, Z', Z'', \ldots$ at further iterations are in superpositions in the negative case. This does not happen in the positive case, where all these registers are in the zero state.

However, all intermediate superposition states lead to *the same* final result. Hence there is no need to investigate these superpositions. This is fortunate, since they look quite "random".

Note 7.4 Since the simulations presented here do not work in constant depth, we still do not know whether $\mathrm{QACC}^k \subseteq \mathrm{QNC}^k_f$, let alone $\mathrm{QTC}^k \subseteq \mathrm{QNC}^k_f$.

7.4 Exact Threshold and Counting gate

Using Theorem 7.6, we can test exactly by the Rank gate in depth $O(\log^* n)$ and space $O(n \log n)$, whether the rank of n source qubits is equal to an arbitrary fixed value m. The rank is the linear combination $z = \sum_{k=1}^{n} \alpha_k x_k$ of the source qubits. We will show next, that the other important gates can be built from an ensemble of these gates.

Let $\{\alpha_k\}_{k=1}^n$ be any integer weights of n qubits. Let $V = \{v_1, v_2, \ldots, v_w\}$ be the set of possible values of the linear combination $z = \sum_{k=1}^{n} \alpha_k x_k$, $x_k \in \{0,1\}$. Let $w = \#V$. For every value $v \in V$, we can test exactly by the linear Rank gate, whether $z = v$.

We can fan-out the source qubits to w copies and then, in the j-th copy, test whether $z = v_j$. All these tests are done in parallel. We obtain w new boolean variables $\{|r_j\rangle\}_{j=1}^{w}$, where $|r_j\rangle = |1\rangle$ iff $z = v_j$, otherwise $r_j = |0\rangle$. For every input, exactly one of these w tests yields true and the others yield false. This costs depth $O(\log^* n)$ and $O(wn\log n)$ ancilla qubits.

Next, we use these intermediate results to compute the desired result and, finally, uncompute everything. Let us look at particular examples.

Theorem 7.8 The linear Threshold gate on n qubits with weights $\{\alpha_k\}_{k=1}^{n}$ upper-bounded by $p(n)$ and threshold m can be computed exactly by a quantum circuit with fan-out of depth $O(\log^* n)$ and using $O\left(n^2 p(n)\log n\right)$ qubits.

Proof. Straightforward, using the described method. Since $|\alpha_k| \leq p(n)$, it follows that $|z| = |\sum_{k=1}^{n} \alpha_k x_k| \leq n \cdot p(n)$. Hence $w = \#V \leq 2n \cdot p(n)$. We fan-out the source variables, compute the boolean variables $|r_j\rangle$, perform a final test, and uncompute everything. The final test consists of a parity gate over the variables $\{|r_j\rangle | v_j \in V \,\&\, v_j \geq m\}$. Since not more than one of these variables is set, the parity operation yields whether $z \geq m$.

The circuit should be optimised in the following way: the boolean variables $\{|r_j\rangle | v_j \in V \,\&\, v_j < m\}$ are not computed, since they are never used.

The circuit uses $O(wn\log n) = O\left(n^2 p(n)\log n\right)$ ancilla qubits and its total depth is $O(\log^* n)$. □

Theorem 7.9 The linear Counting gate on n qubits with weights $\{\alpha_k\}_{k=1}^{n}$ upper-bounded by $p(n)$ can be computed exactly by a quantum circuit with fan-out of depth $O(\log^* n)$ and using $O\left(n^2 p(n)\log n\right)$ qubits.

Proof. Let $b = \log_2\lceil 2n \cdot p(n)\rceil$ be the appropriate size of the register. The goal is to compute exactly the integer value z and store it into the target register $|Z\rangle = |Z_{b-1}\ldots Z_1 Z_0\rangle$.

Again, we first compute the variables $\{|r_j\rangle | v_j \in V\}$. This costs most of the ancilla qubits and it takes $O(\log^* n)$ layers. Next, we fan-out these variables into b copies — each copy is used for another qubit of the register $|Z\rangle$. Finally, we compute the output qubits $|Z_l\rangle$ for $l \in \{0, 1, \ldots, b-1\}$ in parallel and clear ancilla qubits by uncomputation.

The computation of the output qubit $|Z_l\rangle$ is done as follows. We use the boolean variables $\{|r_j\rangle | v_j \in V \,\&\, v_j \bullet 2^l \neq 0\}$, i.e. those whose rank has the l-th bit set. We apply the parity gate on them and store the result into $|Z_l\rangle$. If the l-th bit of z is 1, then exactly one of these tests yielded true, otherwise all of them yielded false. Obviously, the parity gate computes the desired value of $|Z_l\rangle$ and the computation takes another $O(1)$ layers.

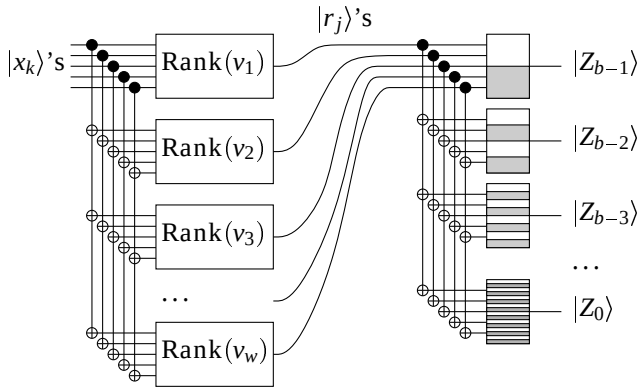

Figure 7.4: The Counting gate built up from Rank gates

The circuit is outlined in Figure 7.4. (Again, only important qubits are highlighted.) The total depth of the circuit is $O(\log^* n)$ and the circuit uses $O(wn\log n + bw) = O(w(n\log n + \log 2n + \log p(n))) = O(wn\log n) = O\left(n^2 p(n)\log n\right)$ ancilla qubits. □

7.5 Arithmetics

Note 7.5 The method from Theorem 7.1 can be also used for performing arithmetical operations. If the source qubits $|x_j\rangle, |y_j\rangle$ control adding 2^j to the register Z (instead of 1) and the size of the register Z is $p = n+1$ (instead of $p = O(\log n)$), we obtain the sum $|x+y\rangle$. The product $|xy\rangle$ can be computed in a similar way — adding 2^{j+k} to the $2n$-qubit register Z is controlled by the logical conjunction $|x_j \wedge y_k\rangle$.

However, from using the QFT on the register Z, the final circuit is approximate and of depth $O(\log p) = O(\log n)$. There exist exact classical circuits of the same depth (the summing is straightforward and the multiplication is a bit tricky). Furthermore, from Theorem 8.1, these operations can be computed exactly by a $O(\log^* n)$ circuit.

Note 7.6 The modular exponentiation (computing $z = a^b \bmod c$) could not be performed by this method. To perform the individual modular multiplications in parallel, we would first have to convert the register Z into a suitable basis. The operator $M_{a,c}$ (multiplying Z by a modulo c) is not diagonal in the Fourier basis. The change of basis seems to be as hard as computing the discrete logarithm.

Chapter 8

Lower and upper bounds

This chapter closes the document. We compare the quantum circuit classes to each other and try to find lower and upper bounds for them.

We first deal with shallow circuits for several arithmetical problems (published in [SBKH93]). They are inferred from neural networks by approximating the Threshold gate by a quantum circuit of depth $O(\log^* n)$.

Furthermore, we present a modified version of Shor's factoring algorithm (published in [CW00]). It consists of classical polynomial time pre-processing and post-processing stages and a logarithmic depth quantum circuit performing the Order Finding. It implies that FACTORING $\in \mathrm{RPF}^{\mathrm{QNC}^1}$. Assuming that FACTORING is not in RPF, it follows that QNC^1 (let alone QNC^1_f) are probably not contained in P. If $\mathrm{QNC}^1 \subseteq \mathrm{P}$, then FACTORING $\in \mathrm{RPF}^P = \mathrm{RPF}$, which is a contradiction.

Finally, we outline relations between the investigated quantum circuit classes and recapitulate open problems.

8.1 Arithmetics using threshold circuits

In [SBKH93], there have been developed shallow threshold circuits for several arithmetical operations. We have already shown that every threshold gate can be simulated by a quantum circuit with fan-out having depth $O(\log^* n)$ and polynomial size. Hence the results inferred in the threshold circuit model can be reformulated also in the model of quantum circuits with fan-out.

Let us just state the results. Look at the original article for a detailed discussion and for proofs.

Definition 8.1 A *linear threshold function* $f(X) : \{0,1\}^n \to \{0,1\}$ is a Boolean function such that $f(X) = \text{sgn}(F(X))$, and $F(X) = \sum_{k=1}^{n} w_k x_k + w_0$ for $X = (x_1, \ldots, x_n) \in \{0,1\}^n$. We assume the weights w_k are integers and bounded by a polynomial in n, i.e. $|w_k| \leq n^c$ for some constant $c > 0$.

A *threshold circuit* (or equivalently a *neural network*) is a Boolean circuit of linear threshold gates. The *depth* and *size* of the circuit is defined in a usual way. We assume the sizes of the considered circuits are all polynomially bounded. We do not impose any restriction on the *fan-in* and *fan-out* of each threshold gate in the circuit.

Theorem 8.1 The product of two input n-bit integers can be computed in a depth-4 threshold circuit.

Theorem 8.2 Let X be an input n-bit integer and let m be a fixed positive integer bounded by a polynomial in n. Then $X \bmod m$ can be computed in a depth-2 threshold circuit.

Theorem 8.3 Let X be an input n-bit integer. Let p be a prime number bounded by a polynomial in n and c be a positive integer not divisible by p. Then $X^n \bmod p$ and $c^X \bmod p$ can both be computed in depth-2 threshold circuits.

Theorem 8.4 Let X be an input $O(\log n)$-bit integer (i.e. $X \leq n^k$ for some constant k) and $c \geq 0$ be a fixed integer. We define EXPONENTIATION to be the $O(n^k)$-bit representation of c^X. The EXPONENTIATION can be computed in a depth-2 threshold circuit.

Theorem 8.5 Let X be an input n-bit integer. We define POWERING to be the n^2-bit representation of X^n.The POWERING can be computed in a depth-4 threshold circuit.

Theorem 8.6 Let $\{X_k\}_{k=1}^{n}$ be n input n-bit integers. We define MULTIPLE PRODUCT to be the n^2-bit representation of $x_1 x_2 \cdots x_n$. The MULTIPLE PRODUCT can be computed in a depth-5 threshold circuit.

Theorem 8.7 Let X and $Y \geq 1$ be two input n-bit integers. Let X/Y be the quotient of X divided by Y. We define $\text{DIV}_k(X/Y)$ to be X/Y truncated to the nearest $(n+k)$-bit number. Then $\text{DIV}_k(X/Y)$ can be computed in a depth-4 threshold circuit.

Theorem 8.8 The SORTING of n input n-bit numbers can be computed in a depth-3 threshold circuit. Any polynomial size threshold circuit for SORTING must have depth at least 3.

Note 8.1 Depths of many presented circuits have been further improved. However, for our purposes, it is only important, that the depths are constant, i.e. the equivalent quantum circuit has depth $O(\log n)$ (if fan-out is available, depth $O(\log^* n)$ is enough). In particular, we are interested in the MULTIPLE PRODUCT problem.

8.2 Factoring problem

In [CW00], there has been discussed a simple modification of the Shor's factoring algorithm. Let us review their method.

The Shor's factoring algorithm works as follows. It is assumed the input is an n-bit integer N.

1. (Classical) If N is even, then output 2. Otherwise determine whether $N = a^b$ for integers $a \geq 1$, $b \geq 2$ (it can be done fast, since there are only $O(\log N)$ possible b's), and if so output the factor a. Otherwise continue to step 2.

2. (Classical) Randomly select $a \in \{2, 3, \ldots, N-1\}$. If $\gcd(a, N) > 1$, then output $\gcd(a, N)$, otherwise continue to step 3.

3. (Quantum) Attempt to find information about the order of a in $\mathbf{Z}_N^*$:

   3.1. Initialise a $2n$-qubit register and an n-qubit register to computational basis state $|0\rangle|1\rangle$.

   3.2. Perform the Hadamard transform on each qubit of the first register.

   3.3. (Modular exponentiation step) Perform the unitary mapping $|x\rangle|y\rangle \to |x\rangle|a^x y \bmod N\rangle$.

   3.4. Perform the inverse QFT on the first register and measure it in the computational basis. Let y denote the result.

4. (Classical) Use the continued fraction algorithm to find relatively prime integers k and r, $0 \leq k < r < N$, such that $|y/2^{2n} - k/r| \leq 2^{-2n}$. If $a^r \equiv 1 (\bmod N)$, then continue to step 5, otherwise the computation has been erroneous (it can be restarted from step 3).

5. (Classical) If r is even and $a^{r/2} \not\equiv -1 (\bmod N)$, then compute $d_{1,2} = \gcd(a^{r/2} \pm 1, N)$ and find out which of them is a nontrivial factor of N. Otherwise the computation has been unsuccessful (it can be restarted from step 2).

Let us assume N is composite. Step 1 ensures that N is an odd integer with more than one prime factor. Step 2 filters out a trivial case of finding a factor casually. It is shown in [NC00] that, in Step 4, the probability of yielding a correct order r is at least $\frac{1}{4}$ (if we use a slightly more complicated algorithm involving 2 repetitions of the order finding procedure). For a correct order r, it holds that r is even and $a^{r/2} \not\equiv -1 (\mathrm{mod} N)$ with probability at least $\frac{1}{2}$. It follows that at least one of $d_{1,2}$ is a nontrivial factor in such case.

Corollary 8.9 If N is composite, then the algorithm finds a solution in one iteration with probability at least $\frac{1}{8}$, otherwise it never returns a wrong answer. Hence FACTORING can be computed by a polynomial time algorithm with bounded one-sided error using a quantum oracle.

The bottleneck of the quantum part of the algorithm is *not* the QFT, but the modular exponentiation. However, as noted in [Sho97], most of the work can be shifted to the classical computation in Step 2 of the procedure. If we pre-compute $b_k = a^{2^k} \mathrm{mod} N$ for $k \in \{0, 1, \ldots, 2n-1\}$ using the recursive formulae $b_{k+1} = b_k^2 \mathrm{mod} N$, then

$$a^x = a^{\sum_{k=0}^{2n-1} 2^k x_k} = \prod_{k=0}^{2n-1} \left(a^{2^k}\right)^{x_k} \equiv \prod_{k=0}^{2n-1} b_k^{x_k} = \prod_{k=0}^{2n-1} b'_k (\mathrm{mod} N),$$

which is an instance of the MULTIPLE PRODUCT problem. The quantum circuit gets all b_k's on the input. If $x_k = 1$, then it sets $b'_k := b_k$, otherwise it sets $b'_k := 1$. The modular exponentiation is then performed by simply multiplying $b'_0 b'_1 \cdots b'_{2n-1}$.

In [CW00], the presented method has led to a result, that the quantum part of the algorithm can be performed by a quantum circuit of depth $\mathrm{O}\left(\log^2 n\right)$. It follows from the fact, that QFT $\in \mathrm{QNC}^1$ and that the $2n$ multiplications can be performed by a binary tree of depth $\mathrm{O}(\log n)$ and each multiplication can be performed in $\mathrm{O}(\log n)$ layers.

This simple solution can be further improved by incorporating the presented shallow circuit for computing the MULTIPLE PRODUCT (having depth $\mathrm{O}(\log n)$, respect. $\mathrm{O}(\log^* n)$ if the fan-out is available). The QFT $\in \mathrm{QNC}^1$ and MULTIPLE PRODUCT $\in \mathrm{QNC}^1$. Hence the quantum part of the computation is also in QNC^1.

Corollary 8.10 FACTORING $\in \mathrm{RPF}^{\mathrm{QNC}^1}$, i.e. it is computable by a polynomial time algorithm with bounded one-side error using the oracle solving QNC^1 problems. The algorithm either returns a non-trivial factor of N, or

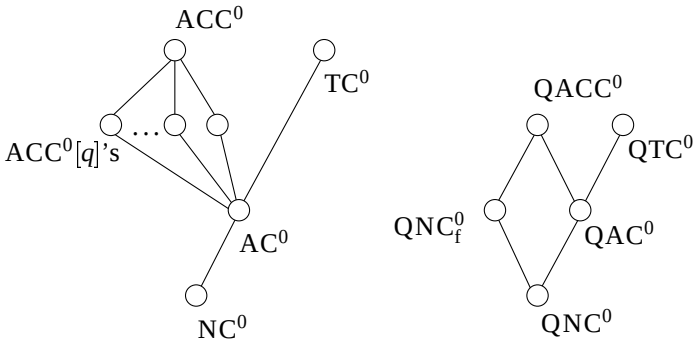

Figure 8.1: Relations of the circuit classes

conjectures that N is a prime number. It can only mistake that a composite number N is proposed to be prime.

Assuming FACTORING $\notin$ RPF, then QNC$^1 \not\subseteq$ P. If it held that QNC$^1 \subseteq$ P, then FACTORING $\in$ RPF$^P =$ RPF, which is a contradiction.

Note 8.2 In the model of quantum circuits with fan-out, the bottleneck of the algorithm is the QFT. If it can be improved to depth $O(\log^* n)$, then the whole quantum part of the algorithm would have depth $O(\log^* n)$.

8.3 Relations of the quantum circuit classes

It is obvious that QNC$^k \subseteq$ QNC$^k_f \subseteq$ QACC^k. Unfortunately it is not known at this time, what the relation between QAC^k, QTC^k, and QNC^k_f is. We have presented several interesting simulations using the fan-out gate:

- The linear Rank gate and thus also the Toffoli gate of size n can be simulated exactly in depth $O(\log^* n)$ using $O(n \log n)$ ancilla qubits.
- The linear Threshold gate and the linear Counting gate of size n with weights bounded by $p(n)$ can both be approximated with polynomially small error in depth $O(\log\log n)$ using $O(n \log n)$ ancilla qubits. They both can be also computed exactly using $O\left(n^2 p(n) \log n\right)$ ancilla qubits in depth $O(\log^* n)$.

However none of these results imply that QAC$^0 \subseteq$ QNC^0_f, let alone QTC$^0 \subseteq$ QNC^0_f. The converse simulation (of the parity gate in the circuit class QAC^0 or QTC^0) is also unknown yet.

8.4 Open problems

We do not know whether $\mathrm{QAC}^k \subseteq \mathrm{QNC}^k_f$, let alone $\mathrm{QTC}^k \subseteq \mathrm{QNC}^k_f$. To give a positive answer, we would have to improve the presented $\mathrm{O}(\log^* n)$ depth circuits to depth $\mathrm{O}(1)$. Hence it would be necessary to get rid of the iteration $n \to \mathrm{O}(\log n)$ and solve the Rank problem immediately in the first iteration. We do not know how to gather more information from the superposition state of the register Z after the Hadamard operation is applied for the second time — the state looks very "noisy".

We do not know whether the QFT can be performed in $\mathrm{o}(\log n)$ depth. The QFS can be performed in constant depth, but it leaves source qubits unchanged, which disturbs the desired interference pattern of the target qubits. The QFP circuit, presented in [CW00], uses a reduction of a sequence of n numbers using an associative operation, hence it can be done by a balanced binary tree in depth $\mathrm{O}(\log n)$. However the binary operation is not commutative, thus we can not directly apply the parallelisation method.

Let V be an operator on n qubits that approximates U with error ε for each computational basis state. We know that the expected error for a random superposition state is also ε, but the maximal error is known to be bounded by $2^{n/2} \cdot \varepsilon$. Could this bound be improved?

The space complexity of the $\mathrm{O}(\log^* n)$ depth Counting and Threshold operators is quite big, since we apply the Rank gate for each possible value $v \in V$ of the variable z. Can it be decreased?

Bibliography

[BBC+95] A. Barenco, C. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52:3457–3467, 1995. quant-ph/9503016.

[CEMM98] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca. Quantum algorithms revisited. *Proc. R. Soc. London A*, 454(1969):339–354, 1998. quant-ph/9708016.

[CW00] Richard Cleve and John Watrous. Fast parallel circuits for the quantum Fourier transform. In *Proc. of the 41st IEEE Symp. on Foundations of Computer Science*, pages 526–536, 2000.

[GHMP02] F. Green, S. Homer, C. Moore, and C. Pollett. Counting, fanout, and the complexity of quantum ACC. *Quantum Information and Computation*, 2(1):35–65, 2002. quant-ph/0106017.

[GN96] R. B. Griffiths and C.-S. Niu. Semiclassical Fourier transform for quantum computation. *Phys. Rev. Lett.*, 76(17):3228–3231, 1996. quant-ph/9511007.

[Moo99] Christopher Moore. Quantum circuits: Fanout, parity, and counting. quant-ph/9903046, 1999.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, U.K., 2000.

[Pre97] J. Preskill. Lecture notes for quantum information and computation course. `http://www.theory.caltech.edu/~/people/preskill/ph219/#lecture`, 1997.

[SBKH93] Kai-Yeung Siu, Jehoshua Bruck, Thomas Kailath, and Thomas Hofmeister. Depth efficient neural networks for division and

related problems. *IEEE Transactions on Information Theory*, 39(3):946–956, 1993.

[Sho94] P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proc. of the 35th Annual Symp. on FOCS*, pages 124–134, Los Alamitos, CA, 1994. IEEE Press. `http://citeseer.nj.nec.com/14533.html`.

[Sho97] P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. quant-ph/9508027.

[Špa02] Robert Špalek. Space complexity of quantum computation. Master's thesis, Faculty of Mathematics and Physics, Charles University, Prague, 2002. `http://www.ucw.cz/~robert/qbp/`.

[Wat98] John Watrous. Relationships between quantum and classical space-bounded complexity classes. In *IEEE Conference on Computational Complexity*, pages 210–227, 1998.