BOSTON UNIVERSITY
GRADUATE SCHOOL OF ARTS AND SCIENCES

Dissertation

QUANTUM CIRCUITS: POWER AND LIMITATIONS

by

DEBAJYOTI BERA
B.Tech., Indian Institute of Technology, Kanpur, 2002

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
2010

Approved by

First Reader

Steven Homer, Ph.D.
Professor of Computer Science
Boston University

Second Reader

Frederic Green, Ph.D.
Professor of Mathematics and Computer Science
Clark University

Third Reader

Péter Gács, Ph.D.
Professor of Computer Science
Boston University

DEDICATION

To *Bhai*

and

other young minds of Kolkata

Acknowledgements

"So long, and thanks for all the fish"

Looking back at the year 2001, I see a lean, bespectacled IIT-ian madly hitting a keyboard to fix the X configuration of some friend's Linux installation. It was almost decided, by my peers of course, that the next protagonist of the *Open source movement* would be "dBera". That disaster was narrowly, and thankfully, averted when I enrolled for the "Circuit Complexity Theory" course taught by Prof. Manindra Agrawal. Many of the random ideas that struck me during that course took shape in various forms over the next several years and culminated in this thesis. Nostalgia breaths even more because one of the next chapters is based on a question that first occurred to me back then.

Prof. Agrawal deserves a special mention because other than being an excellent teacher, he was also my undergraduate advisor and introduced me to "Quantum Computing" as part of my final year project. He is solely responsible for turning my attention towards theoretical computer science. This thesis, on quantum circuit complexity, is my humble token of appreciation for his effort and sincerity.

Barring a sojourn of 6 months, the cruise continued on unknown waters, with a different commander now on the deck. My Ph.D. advisor, Prof. Steve Homer, patiently let me explore the depths but gave the screw the crucial half-turn just when needed to prevent me from floating away. I can't thank him enough for his constant encouragement and his confidence in me.

My co-advisor, Prof. Fred Green, has a direct contribution in this thesis with his stylish penchant for precision and the quotation in the final chapter. His overall effect on my research spans a multitude of expert guidance, from technicalities of circuit complexity to

physics insights to quantum computing. Anytime I had a crazy idea, I could safely throw it at him and hope to receive a surprisingly inspiring reply, or at the very least, a witty and funny remark to end the discussion.

I had many fruitful discussions with Prof. Jin-Yi Cai when he was visiting Boston, some of which helped me shape the lower bound results in this thesis. I would also like to thank Prof. Steve Fenner, one of my co-authors, for a successful collaboration on the "universal circuit" paper. I would like to thank Prof. Péter Gács foremost for agreeing to read my thesis and also for all the insights he shared with me at other times. I still have fond memories of one of my first graduate courses in BU which was taught by Péter. Prof. Leo Reyzin, who agreed to be on my thesis defence committee, exposed me to the dark world of thieves and eavesdroppers (aka cryptography). Now I not only find cryptography research stimulating but I also know how to keep myself secure [1].

I could not have asked for a better mentor than my long time office-mate Benjamin "Joe" Hescott. Soon after we started sharing office during my second semester, he began my training for initiation into the league of *extraordinary graduate students*. Over the next few years, I saw first-hand how to explore *The Complexity Zoo*, set goals for research and go about achieving them. I believe his training was successful, as a sign of which, he bestowed me with a great many bounties when he left, including a well-maintained bike.

All that I inherited from Ben and still unconsumed, will now be passed on to Bhavana Rukmini Kanukurthi, my office-mate and the next care-taker of the theory cells. After Ben left, she occupied his desk and along with that, the hat of a perfect office-mate anyone could ask for. It is not uncommon during graduate study to slip and lose sight of the goal. She kept an unerring eye on me to ensure an unhindered finish and offered support, asked or unasked, professional and personal. Befriend her if you get a chance, if only for the delicious *aloo-parathas*[2] and *chai* that she makes.

Kyle "Monkey"[3] Burke began after me but beat me in graduation. He actually made it

[1] (Off the record) and, how to steal without getting caught.
[2] Not many Indian restaurants in the USA serve this stuffed flat-bread delicacy.
[3] self-proclaimed

easier for me by figuring out the logistics. Always overflowing with energy, he is a great asset for any gathering. We were partners in crime (nothing serious here). I am greatly indebted to him also for his ... the list is long but I don't have the space. Sorry, Kyle!

Scott was the quintessential American, and graced our tiny bench of culinary experts. Pastry and workout, party and sincerity, these are allowed to appear together when we talk about him. During his final year, his job was to keep my procrastination under control and I am sure I would have graduated earlier if he deferred his. The last person from the *older generation* who was a part of the intellectual hangout of *PSY-221* is Nenad. It was, and is, a pleasure talking to him, maybe because he knows how to read between my lines or maybe because he plays computer games. I interacted with both Scott and Nenad on some of their crypto research, and I owe a large part of my current skills in cryptography to them.

After Ben, Scott and Nenad ditched my words of wisdom in exchange of real paychecks, the reputation of our office was kept floating by Ilir and David (besides Bhavana and Kyle). I would like to officially thank the former, a diver from a small country, for marrying someone who makes über-awesome Turkish coffee. She's the best. David was always the oracle, especially in algebra, with ready solutions to most of the questions we asked. When I was his TF, I surely learnt more Algebra myself than the students.

I made many new friends in this department, and some of the friendship will hopefully continue even after I leave. The faces that immediately come to the forefront are mostly from the networking group. There was a time when I used to spend more time in the network research area (the dual-core dual-monitor machines were just a good excuse). The enthralling atmosphere created by Kanishka, Dhiman, Georgios, Niky and others stimulated a lot of discussions and produced interesting research. It felt great to participate in solving networking research problems without actually writing any code! I always enjoyed my special rapport with Vijay. After he graduated, the lab never felt the same and I miss our long sessions with XBox or on contemporary politics. The camaraderie among the graduate students in the department is something we are proud of.

Surrounded by Ashwin, Gabe, Mike, Vitaly (and his little friend), Sowmya, Vatche and many others, we always had a great time together whenever we met. It was wonderful to have all these great researches as my peers during my graduate days.

Outside of BU too, many new bonds were forged and some existing ones were tightened. Most of my friends outside BU have some form of MIT connection. I became very close friends with Saikat and Siddharth, which is something I will cherish for the rest of my life. Living in Boston would have been very different, and very difficult for sure, if not for the *MIT mafia!*.

On our mission to scale the heights that are set for us, we sometimes share our road with other peregrines, in a way that changes forever our perception of the mission. I was lucky our paths crossed during the final years of my PhD programme. This effort would not have its shine without her words of inspiration, her confidence in me, her endurance, and a lot of simple wishes.

I am dedicating this dissertation to my loving brother, Debarun. While I was busy balancing my way on a slippery road, he took over on his young shoulders the huge responsibility of everything else back home. He sacrificed much to ensure that I have a free run with my apocalyptic idiosyncrasies.

My parents are above and beyond all thanks that I can ever gather. It is plain futile to even try to put into words their support and encouragement for me during all these years as a graduate student. I hope I somewhat succeeded in meeting their expectations.

This thesis, which contains whatever outcome of my research efforts that I could possibly "write"[4] in words, is a joint fruit of labour, persistence and confidence of a lot of people, spread all around the globe. I took this opportunity to thank all those who helped me turn a possibility into a reality. I surely missed more names than I remembered to mention above. But mentioned or unmentioned, my gratitude transgresses the words I used to express my heartfelt thanks.

[4]Though this thesis was typed, all the results were first proved on rough papers, the unsung heroes of academic life.

QUANTUM CIRCUITS: POWER AND LIMITATIONS

(Order No.)

DEBAJYOTI BERA

Boston University, Graduate School of Arts and Sciences, 2010

Major Advisor: Steven Homer, Professor of Computer Science

Boston University

ABSTRACT

Quantum computation was proposed as an alternate computing process in the late-20th century, when classical computing models faced limitation in solving quantum mechanical problems. Major breakthroughs came in the 1990s, when quantum algorithms for integer factoring and black-box searching were invented that are faster than the best classical algorithms known even today. In this thesis, we look at quantum circuits with limited depth and size and analyse their power and limitations.

Computer Science is about the study of computational problems, many of which are motivated by real-world applications. We are interested as much in an actual method to solve a problem as in its unsolvability. In computational complexity theory we separate out the actual problem from the computing apparatus (e.g. a Turing machine, an AND-OR circuit) and analyse the hardness of the computation, due both to the intrinsic complexity of the problem and to the underlying model.

In this thesis, we consider quantum circuits built with a universal set of gates and show that non-constant depth is required to compute a particular function (parity) and other similar functions. The parity function, extensively studied previously, was shown to take non-constant depth for classical circuits too, thus providing an evidence of an inherent hardness in the function itself. On the other hand we show that the quantum circuit model is rich enough to build a universal circuit (one circuit that can simulate all circuits) that has roughly the same depth or size as the simulated circuit. We also consider a fault model in which some of the inputs to a gate might not be connected to the gate's output. We ask if we can detect the faulty inputs by testing the gate on different inputs.

Our goal is to minimise the number of tests. We show that this is very hard for general gates but for certain gates it can be provably easier to detect faulty inputs. We even show that quantum algorithms can provably outperform classical algorithms for these gates. These results give us a sense of wherein lies the power of quantum computation and quantum circuits in particular.

Contents

List of Tables

List of Figures

List of Abbreviations

AC^k . . . . . Poly-size Poly-log Depth Unbounded Fanin Classical Circuit

BQP . . . . . Bounded Error Quantum Polynomial Time

NC^k . . . . . Poly-size Poly-log Depth Bounded Fanin Classical Circuit

NP . . . . . Nondeterministic Polynomial Time

P . . . . . Polynomial Time

PP . . . . . Probabilistic Polynomial Time

QAC^k . . . . . Poly-size Poly-log Depth Unbounded Fanin Quantum Circuit

QAC^k_{wf} . . . . . QAC^k With Fanout

1 Introduction

"There is no quantum world. There is only an abstract physical description. It is wrong to think that the task of physics is to find out how nature is. Physics concerns what we can say about nature..."

– Niels Bohr

It can be debated whether the goal of physics is to understand how nature is or how nature behaves. In Computer Science, our goal is rather modest: To analyse the myriads of *computations* that we experience in nature[1]. Often that process involves analysing other computations that bear little or no similarity to anything that actually exists. These mostly serve to enrich our understanding of the properties of the computations. Since our goal is to gain insight about certain processes, we will henceforth ignore the issue of immediate relevance with the real world. If you are still hesitant to accompany me in this pursuit of theoretical knowledge, bear in mind that the magic number *0* was invented to understand arithmetic and possibly without any industrial application in sight.

Once we view a physical process as some kind of computation, involving an initial state, a time-dependent evolution function and (optional) inputs from outside, everything looks like a nail for this hammer. Pretty much anything around us is a computation in a sense; from making the perfect espresso to finding free food in a university campus[2].

[1] The name *Computer Science* is a misnomer in that sense. But we will not bother ourselves with that here.

[2] This is my modest attempt to acknowledge coffee, which is very essential for great research. As Alfréd Rényi once said, "... a mathematician is a machine for converting coffee into theorems". Free food is another very important resource for graduate students.

One of the first steps in analysing a process is to come up with a suitable model that adequately characterises it. A marvellous exercise in this was undertaken by Antoni Gaudí, a great Spanish architect, who lived around the turn of the century. His design of the holy family church, (*la sagrada familia*) in Barcelona is a masterpiece of art, and is still in the process of being build, after almost a hundred years. The church resembles a sand palace, with a tremendous complexity of delicate thin but tall towers and arcs. Since the plan of the church was so complicated, towers and arcs emerging from unexpected places, leaning on other arcs and towers, it is practically impossible to solve the set of equations which corresponds to the requirement of equilibrium in this complex. Instead of solving this impossible task, Gaudí thought of the following ingenious idea: For each arc he desired in his complex, he took a rope, of length proportional to the length of the arc. He tied the ends of one rope to the middle of some other rope, or where the arcs were supposed to lean on each other. Then he just tied the edges of the ropes corresponding to the lowest arcs, to the ceiling. All the computation was instantaneously done by gravity! The set of arcs arranged itself such that the whole complex is in equilibrium, but upside down. Everything was there, the angles between the different arcs, the radii of the arcs. Putting a mirror under the whole thing, he could simply see the design of the whole church![Aha99].

The above example is lacking in important details e.g. on why a stable structure can also be stable when physically turned upside-down but nevertheless it inspires us to try to capture the relevant details of any complicated problem in a simpler model. Considering the suitable model is very essential for a computation and at the core of any algorithm to solve a problem. This has resulted in many models that are studied in Computer Science. The most common are random access machines, Turing machines, Boolean circuits, cellular automata [vEB90]. Each of these models have some nice feature that makes it suitable for analysing a particular kind of computation. This huge crowd of models would be overwhelming if not for equivalence theorems between the different models. These enable us to continue working in any preferred model yet know that at

the end of the day the result is true irrespective of the model used.

Not all of these models punctiliously follow the computation process it was designed for. Real world processes can be too complex to be captured by simple rules of mathematics and logical deductions. Some of the models, e.g. the non-deterministic models, do not have any practical analogue. Instead, these models strive to capture some of the fundamental characteristics of the computation process and in this way, pave the way for a rigorous analysis of the *complexity* of the computation. In *Complexity Theory*, we separate out the hardness which is due to the model from the inherent model-agnostic difficulty of the problem itself and study them independently.

1.1 Quantum Computing

Quantum computation was first proposed in the late-20th century as an alternate computing process when classical computing models (which were based on Newtonian mechanics) were unable to explain all observed phenomena. Stated simply, during a quantum computation the data is carried by *objects* which follow the laws of quantum mechanics. For example, the spin of an electron might be used to denote 0 and 1 and flipping its spin will mean flipping a bit. Great progress has been made in quantum computation. It is currently even used in the communication industry, especially for secure communication. For a history of quantum computation, see [NC00].

In quantum complexity theory, we try to model the quantum computation with an aim to understand the *magic* that makes everything work. Major breakthroughs came in quantum computing in the 1990s where it was shown that integer factoring [Sho97] or black-box searching [Gro96] could likely be done significantly faster on a quantum computer than on a classical computer. It is to be noted that a practical implementation of quantum integer factoring would cause major disruption in the secure communication and e-commerce industry.

A quantum circuit is one of the ways to model a quantum computation[3]. Some of the other models are quantum Turing machine, quantum decision trees etc. These models have been shown to be roughly equivalent to one another. We will describe the details of the quantum circuit model in Chapter 2. But roughly, a quantum circuit consists of a collections of quantum gates which are connected by wires. The wires, like those in classical digital circuits, carry 0 or 1, except that the value can also be a *quantum superposition* of 0 and 1. For now think of the superposition as the state akin to Schrödinger's cat [Wikb] - a wire can simultaneously carry the value 0 and 1 until it is explicitly measured by a physical measuring device. It is profoundly different than the wire carrying 0 or 1 without us knowing which one, in a sense it carries *both* the values. This leads to the funny situation, which is also responsible for the magic, where two wires taken together might not carry a particular value, say 00, with positive probability even though individually the wires could be carrying the value 0 with non-zero probabilities.

Like classical circuits, there are quantum gates that change the values of the wires that pass through them following the rules of quantum mechanics. The quantum circuits that we study are limited in the sense that there are no feedback loops and the number of gates is bounded above by a fixed polynomial in the number of inputs. One or more wires are specified as the output wire; the final value of the circuit is the value obtained by measuring these wires.

1.2 Outline and main contributions

Most of the common models of quantum computing have been proved to be equivalent; this usually means that if a problem can be solved with reasonable resources[4] in one model, then it can also be solved using reasonable resources in another model. However, some peculiarities of the circuit model make it suitable for certain problems. In this thesis,

[3] To other fields e.g. physics, our models might seem too simplistic.

[4] An important resource for a Turing machine could be the number of states or the time taken; for a circuit it could be the number of gates used or the types of the gates.

we try to analyse this model and specially focus on the role of the different gates that are used in the quantum circuits. We provide key insights on wherein lies the strength of quantum circuits and what limits their power. We begin by giving a brief description of the quantum circuit model in Chapter 2.

1.2.1 Universal Quantum Circuit

One of the novelties of modern day computers over earlier computing devices was the idea of stored programs and universal machines. Unlike the calculators which were specialised to perform only a single operation, a computer is designed as a general purpose machine which takes two inputs, an encoded form of the steps of an operation known as the *program* and the *data* for the program. It then executes the program on the data and outputs the output of the program. Such a simulator which can conceptually simulate many different kinds of operations is usually called a universal machine.

There are several remarkable results showing how to construct universal Turing machines and universal machines for other models of computing. In Chapter 3, we explore the idea of a single quantum circuit to simulate a family of similar quantum circuits. We also focus on the efficiency of their simulation. For example, a machine which takes exponential time to simulate a constant step algorithm might not be very interesting. Since the quantum circuit model of computing is equivalent to the Turing machine model, it is obvious that there exists a universal circuit. However, it is not clear how large this circuit is. Valiant, Cook, Hoover and others have studied classical universal circuits. We similarly define quantum universal circuits and give explicit constructions where the universal circuit uses similar resources compared to the simulated circuit.

In Theorem 3.3.2, we show how to construct a quantum universal circuit with only a constant factor increase in depth. We show how to construct a quantum universal circuit with only a minor blowup in the size in Theorem 3.4.5.

1.2.2 Lower Bounds

After we show that low depth (or size) circuits with certain sets of gates can be powerful enough to contain a universal circuit for all circuits of the same depth (or size), we can ask a contrasting question: How powerful is a circuit with a fixed set of gates and of a limited size or depth? Historically, the study of (classical) circuit lower bounds gained popularity as a possible approach to separating P from NP. While progress towards this goal has been slower than as expected, the field of circuit lower bounds has nevertheless remained popular in complexity theory.

In Chapter 4 we look at lower bounds of quantum circuits. Specifically, we consider circuits with constant depth[5] and created by a fixed family of 1-input gates and a specific multi-input gate. Our goal is to prove that this kind of circuits are not powerful enough to compute some functions that we believe to be computationally hard.

Similar questions had been asked before for classical circuits. But quantum circuits often behave in a way that defies intuition. For example, a very useful technique used in classical circuit construction is to take a wire, split it using a wire-splitter or solder another wire to it to obtain several wires carrying an identical value. This obviously simple operation, known as *cloning*, is not allowed in any quantum circuit by the rules of quantum mechanics.

Classical circuits and gates are usually analysed based on the function they compute i.e. the mapping between input and output values. On the other hand, quantum models are inherently probabilistic in nature. For example, for every input, instead of a single output value, there is a probability distribution of outputs. This will be explained in detail in Chapter 2. Thus, the classical techniques cannot be applied anymore. We provide a new lower bound technique using probabilistic analysis of the functions computed by the gates and the circuits (Theorem 4.4.7). This gives us the desired lower bounds for a certain class of quantum circuits. We also apply the technique to show that parity cannot be computed by a certain type of constant-depth quantum circuits (Theorem 4.5.2).

[5]Depth of a circuit is the maximum number of gates from the output to any input.

1.2.3 Detection of Faulty Gates

In any office, tracking a lost file can be significantly easier if the bureaucracy has very strict rules of operation. It is debatable if moving to electronic data would reduce the chances of losing files in the first place but I think most would agree that if a system is very robust against disturbances, then any discrepancy might be easily detected.

In Chapter 5, we consider a new fault model for classical and quantum gates where a faulty gate might not depend on all its inputs. If we are given multiple copies of the same exact faulty gate, then we can always create a circuit using these gates to determine which inputs are faulty. Our goal is to minimise the number of copies of the gate. We consider a few types of gates and show matching classical and quantum upper and lower bounds on the number of such gates that we need to use to detect faults. We also observe that for gates that are harder to compute (this notion will be elaborated in later chapters), it is easier to find these faults; this should be seen as a supporting evidence of the intuition described above.

We consider gates computing the parity function and give a quantum algorithm for faulty quantum parity gates (Theorem 5.4.1) which is not-only optimal for the quantum case but also provably more efficient than any classical algorithm for faulty classical parity gates (Corollary 5.4.4). We similarly consider gates computing the AND function: we give an algorithm (Theorem 5.6.1) for quantum gates computing this function and show it is optimal (Corollary 5.6.2) and provably better than any classical algorithm for faulty AND gates (Corollary 5.6.3).

1.2.4 Future Direction

In this thesis we consider a few fundamental quantum gates and evaluate their properties, especially their relative power against one another. Also, the circuits we consider mostly belong to a specific class. A lot of questions remain unanswered, mostly regarding other types of quantum gates and using different classes of circuits. We end this thesis with

Chapter 6 where we describe what we think are prospective directions for future research in this area.

2 Quantum Circuit Complexity

"Anyone who is not shocked by quantum theory has not understood it."

– Niels Bohr

In this chapter we will present some background on quantum circuit complexity. A full review of quantum computation is beyond the scope of this thesis. For a quick introduction, we recommend Fenner [Fen03a] or Fortnow and Rogers [FR99]; for an in-depth treatment, see Nielsen and Chuang [NC00]. Instead, we will focus on quantum circuits and accordingly introduce the technical concepts needed to understand this thesis. We will also introduce the notation used in the forthcoming chapters. A significant part of this chapter was published in the article:

•[BGH07] D. Bera, F. Green, S. Homer. Small depth quantum circuits. SIGACT News, 38(2), 2007.

2.1 Background

A quantum circuit is similar to a classical Boolean circuit; both can be viewed as an acyclic network of gates interconnected by wires. First we describe general classical Boolean circuits and then introduce quantum circuits by pointing out the differences. In the most general setting, a classical Boolean circuit has certain *input gates* and *output gates*. The circuit is evaluated on a particular input; the input values (bits) are hardwired as the

output of the input gates. The circuit works by evaluating each gate in succession. The output of a gate A could be connected by a wire to another gate B; the output of A serves as the input to B. After all the output gates have generated some value, the output of the circuit is defined as the collection of the values generated by all the output gates. The circuits we consider in this thesis have acyclic graphs as their underlying topology i.e. if the input is fixed, the output of every gate is also fixed. Though we do not consider them here, circuits whose graph contains cycles are a major area of research. In these circuits, the output of a gate at one instant acts as a feedback to the gate at a later instant; they are handled very differently from conventional circuits.

For a classical Boolean circuit with n input gates and m output gates, it is said to compute a Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}^m$ if the output of the circuit if $f(x)$ whenever the input to the circuit is $x \in \{0,1\}^n$. In complexity theory, we are interested in investigating the computational properties of a circuit computing a certain function e.g. its size and depth. The *size* of a circuit is the number of gates in it. The *depth* of a circuit is the number of the gates in the longest path from any output node to any connected input node. Informally, our goal is create circuits with as few gates and as shallow as possible.

The common gates used to construct a Boolean circuit are the AND, OR and NOT gates. Another gate that is sometime used is the *parity* gate computing the Boolean XOR function. The NOT gate always takes one input but the other gates, depending on the type of the circuit, can take a fixed or a variable number of inputs. Usually the classical Boolean gates have one output but the output wire can be split into multiple wires to produce multiple copies of the same output. The number of inputs of a gate is known as its fanin and the number of outputs is known as its fanout. We say that the fanin of a gate is *bounded* if it is bounded above by a constant irrespective of the number of inputs to the circuit; otherwise, we say that the fanin of this gate is *unbounded*. Gates with unbounded fanin are provably more powerful than the gates with constant fanin. Note that for classical circuits, there is usually no restriction on fanout.

2.1.1 Motivation

For many of us studying the quantum computing models, quantum circuits are the most natural and general formulation of quantum computation. They are general in the sense that they are a universal model; any quantum computation can be efficiently simulated by a quantum circuit [Yao93, NO02]. They are natural in that a quantum computation (or quantum algorithm) can best be understood as a collection of qubits being acted on by quantum operators represented as (defined by) a tensor product of quantum gates. Although the quantum circuit model is quite different than the classical one, it has nevertheless proven to be quite fruitful to look to classical circuit models for insight.

For classical circuits, those with small depth (e.g., polylog as in the class NC) have been proposed as realistic models of parallel computation. Furthermore, *shallow* (e.g., constant) depth classical circuits present us with computational models for which we can actually prove interesting lower bounds. What happens when we extend these concepts to quantum circuits?

The answer is a bit surprising, and leads fairly quickly to interesting variants of the fundamental problems of quantum computing. For example, consider the class AC^0 of constant-depth, polynomial-size classical circuits consisting of NOT together with unbounded fan-in and fan-out AND and OR gates. Some interesting functions can be computed in this class (addition of n-bit numbers for example), and some cannot (e.g., parity [FSS84]). It would be interesting to see what an analogous quantum complexity class would look like. But if we try to translate the class AC^0 into the quantum setting, we are faced with an immediate problem. The unbounded-fanin "quantum AND gate," the generalised Toffoli gate (which also encompasses negation and hence OR as well) does not allow for fanout in any way we may take for granted. The reason is the "no-cloning theorem," which says that it is in general not possible to make a copy of a quantum state. One can, however, make copies of classical bits, which suggests the idea of introducing a unitary operator to implement fanout of classical bits. (A similar operation is really implicit in the AC^0 model; we solder multiple outgoing wires to an AND gate, obtaining

copies of the output. While it is not entirely realistic to have an unbounded number of wires fanning out of the gate even classically, this is a useful abstraction out of which the definition of AC^0 arises.)

We therefore seem to have two candidates for quantum analogs of AC^0. One, which just includes generalised Toffoli gates and single-qubit gates, is called QAC^0. The other, which includes fanout gates, is called $\mathsf{QAC}^0_{\mathrm{wf}}$. The latter appears to be the more realistic version, since it is straightforward to see that it includes AC^0. However, we will point out in later this chapter that $\mathsf{QAC}^0_{\mathrm{wf}}$ is much more powerful than AC^0. There nevertheless remains much that we do not understand. For example, it is unknown to this day if QAC^0 is the same as $\mathsf{QAC}^0_{\mathrm{wf}}$.

This gives an interesting theoretical framework in which quantum analogues of classical circuit models are provably more powerful. For our implementation-minded readers, does all of this have any bearing on reality? It might. Realisable quantum computations will have very limited duration, due to short coherence times, which suggests that highly parallelised quantum circuits (such as those we will use in Chapter 3) are desirable. Fanout gates as well as Toffoli gates might actually be feasible to build via ion trap [CZ95] or bulk NMR techniques [GC97]. It may therefore be of more than mere theoretical interest to explore their power, both intrinsic and relative to each other.

2.2 Quantum Circuit Model

Structurally, a quantum circuit differs from a classical Boolean circuit in two basic ways; instead of bits, *qubits* flow[1] through the wires and get transformed by quantum gates. Like a bit, a qubit can be in the state 0 or 1; additionally a qubit can also be in a superposition of 0 and 1 at the same time. Mathematically, a qubit is a vector in a two-dimensional Hilbert space $\mathcal{H}$; for the purpose of this thesis (and in general for quantum computing),

[1] Classically, a bit might represent the level of voltage. Similarly, a qubit might correspond to the spin of an electron or the polarity of light. It should be kept in mind that due to the wave-particle duality nature of the quantum carrying elements, a qubit might not physically *flow* through the wires.

$\mathcal{H}$ is same as a vector space defined over complex numbers with following inner product:

$$\text{inner product of } (x_1, x_2), (y_1, y_2) = x_1^* y_1 + x_2^* y_2$$

We follow Dirac's *bra-ket* notation to represent a qubit by $|q\rangle$ and the two basis states of a qubit will be denoted by $|0\rangle$ and $|1\rangle$. Qubits q and s are orthogonal if their inner product, denoted by $\langle q|s\rangle$ is zero. The norm or length of a qubit is defined by

$$\||q\rangle\| = \sqrt{\langle q|q\rangle}$$

The standard basis states are orthonormal. Unless explicitly mentioned, in this thesis, we will be dealing with qubits of length 1.

Let $\mathcal{H}_1, \ldots, \mathcal{H}_n$ be n copies of $\mathcal{H}$. By $\mathcal{B}_n$ we denote the 2^n-dimensional Hilbert space $\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_n$ spanned by the usual set of standard basis states of the form $|x_1\rangle \otimes |x_2\rangle \otimes |x_3\rangle \otimes \cdots$ (sometimes shortened to $|x_1, x_2, x_3, \cdots\rangle$) each $x_i \in \{0, 1\}$. A "state over a set of n qubits" is a state in $\mathcal{B}_n$, written as a superposition over these basis states.

It is easier to think of a classical circuit as a network of gates connected by wires along which bits travel only along one direction; they enter a gate which subsequently outputs a single bit. To understand the operations of a quantum circuit, consider this alternative perspective of a classical circuit. Rearrange the gates of the circuit in layers and represent the *state* of the circuit by the values in the wires in that layer. The gates between two layers are now responsible for the collective change of state between these layers. A quantum circuit can be similarly viewed as a series of layers of gates that change the state of the wires between its layers.

For a quantum circuit, a general state over n qubits is usually represented by the elements (also known as the *amplitudes*) of a 2^n-dimensional vector in a suitable basis state. Unless otherwise specified, we will always use the standard basis for representing any state. For example, a state $|\Psi\rangle$ over 2 qubits, in the standard basis of

$|0,0\rangle, |0,1\rangle, |1,0\rangle, |1,1\rangle$ could be the following vector

$$\begin{bmatrix} \frac{1}{4} \\ -i \\ \frac{1}{\sqrt{2}} \\ \frac{1+i}{10} \end{bmatrix}$$

and is usually written as

$$|\Psi\rangle = \frac{1}{4}|0,0\rangle + (-i)|0,1\rangle + \frac{1}{\sqrt{2}}|1,0\rangle + \frac{1+i}{10}|1,1\rangle$$

Quantum mechanics enforces that the state of a group of qubits can be changed by either *applying a quantum gate* or *performing a measurement*. Let $\mathcal{U}_n$ denote the set of unitary matrices that act on the states in $\mathcal{B}_n$ ($\mathcal{U}_n$ is just a convenient notation for the group $U(2^n)$ of $2^n \times 2^n$ unitary matrices). An n-qubit quantum gate G corresponds to an element of $\mathcal{U}_n$, that changes the state of n qubits when operated on. Thus, for example, a *single-qubit gate* is an element of $\mathcal{U}_1$, acting on states in $\mathcal{B}_1$. Often we will write $G|q_1, \ldots\rangle$ to denote the state of a group of qubits after a gate G is applied. Due to the properties of a unitary transformation, a quantum gate preserves the norm of a state and is reversible. We will use $\||q\rangle\|$ to denote the L_2 norm of the vector corresponding to q, which we will also refer to as the length of the state.

On the contrary, a measurement involves projecting the state of a quantum system on a set of basis states; after measurement the measured qubits end up in one of the basis states with probability equal to the length of the projected state along that basis state. At the end of a computation in a quantum circuit, the group of output qubits are usually measured in the standard orthonormal basis. This is a very simplistic view of measurement in a quantum circuit and should be enough to understand the quantum measurements we will consider in this thesis. In general, quantum measurements are defined to be a collection of operators $\{M_m\}$ where m represents the outcome of the measurement, such that $\sum_m M_m^\dagger M_m = I$; after applying the measurement operators to a state $|\Psi\rangle$, the value m is observed with probability $\|M_m|\Psi\rangle\|$. Please refer to [NC00]

for the details on quantum measurements. Observe that this differs significantly from classical circuits because now the output of a circuit is not a fixed value but a probability distribution over a set of possible output values.

Usually the initial input to a circuit is a (pure) state with unit length. Since any unitary operator preserves length, as long as no measurement is made, all along the circuit the qubits are in a state of unit length. We do not allow any intermediate measurement in the circuit model we consider in this thesis. To give an example of a measurement, consider the following state on 2 qubits: $|\Psi\rangle = a_{00}|0,0\rangle + a_{01}|0,1\rangle + a_{10}|1,0\rangle + a_{1,1}|1,1\rangle$. When it is measured in the standard basis, we will observe the measured qubit to be $|i,j\rangle$ with probability $|a_{ij}|^2$.

2.2.1 Quantum Gates

Listed below are some of the important single qubit quantum gates (Figure 2.1). We also show their representation in the standard basis in Figure 2.2.

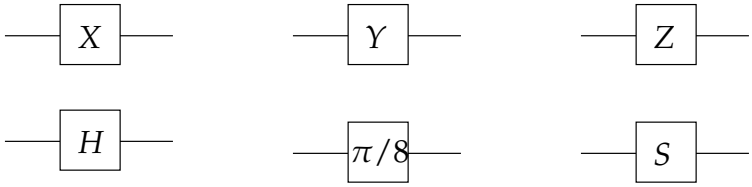

Figure 2.1: Some important single qubit gates: Pauli X gate, Pauli Y gate, Pauli Z gate, Hadamard gate, $\pi/8$ gate and phase gate

$$X = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \qquad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$
$$H = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \qquad \pi/8 = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \qquad S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$

Figure 2.2: Matrix representations of the popular quantum gates

The action of any quantum transformation can be represented very conveniently using the bra-ket notation. In this thesis we follow this notation. As an example, we show below how we represent the action of 3 of the above gates in the bra-ket notation. For $x = 0, 1$,

Hadamard gate: $|x\rangle \rightarrow \frac{(-1)^x}{\sqrt{2}}|x\rangle + \frac{1}{\sqrt{2}}|1-x\rangle$

ß/8 gate: $|x\rangle \rightarrow e^{ix\pi/4}|x\rangle$

Phase gate: $|x\rangle \rightarrow e^{ix\pi/2}|x\rangle$

The H gate is very useful and it will appear in various places throughout this thesis. One of its many useful properties is that it is its own inverse. Specifically, $H^\dagger = H$, where $U^\dagger = (U^T)^*$ is the Hermitian conjugate or adjoint of any matrix representation of a unitary operator U. Before we move to the gates with larger fanin, we want to point out another very useful gate, the two qubit CNOT gate, which performs the following transformation on basis states: $|x\rangle|y\rangle \rightarrow |x\rangle|x \oplus y\rangle$. Note that this gate can be used as a NOT gate by setting y to 1.

Multi-qubit gates Let $f : \{0,1\}^n \longrightarrow \{0,1\}$ be a Boolean function of n inputs. Many of the commonly used multi qubit gates take the form,

$$G|x_1, ..., x_n, b\rangle = |x_1, ..., x_n, b \oplus f(x_1, \ldots, x_n)\rangle \tag{2.1}$$

This is just an easy form of simulating any classical function in a reversible manner. For such gates, the qubits whose value remains unchanged are referred to as the *control* qubits (the first n qubits in Eq. 2.1 above) and the qubit whose value changes is called the *target* qubit (the last qubit in Eq. 2.1).

If $f(x_1, \ldots, x_n) = \wedge_{i=1}^n x_i$, G is called a *generalised Toffoli gate*, or (in this thesis) simply a *Toffoli* gate, and is written as T. If $f(x_1, \ldots, x_n) = \bigoplus_{i=1}^n x_i$, G is a *parity* gate, written P. Observe that the CNOT gate is same as the Toffoli gate for 2 qubits. Generalising this, define the classical Boolean function $\text{Mod}_q : \{0,1\}^n \longrightarrow \{0,1\}$ so that $\text{Mod}_q(x_1, \ldots, x_n) = 1$ iff $\sum_{i=1}^n x_i \not\equiv 0 \pmod q$. If $f = \text{Mod}_q$, we call G a MOD_q gate. Next, for any fixed t, define the Boolean function $s(x_1, ..., x_n) = 1$ iff $\sum_{i=1}^n x_i \geq t$. If $f = s$, we call G a *threshold gate*.

A Z *gate* is an extension of the single-qubit Pauli Z gate ($|x\rangle \mapsto (-1)^x|x\rangle$) to multiple qubits. A Z gate does not any control qubits or target qubits.

$$|x_1, \cdots, x_n\rangle \overset{Z}{\mapsto} (-1)^{x_1 x_2 \cdots x_n}|x_1, \ldots, x_n\rangle.$$

Similarly, a *Z-fanout gate* Z_n applies the single-qubit Z gate to each of n target qubits if the control qubit is set:

$$|c, t_1, \cdots, t_n\rangle \overset{Z_n}{\mapsto} (-1)^{c\cdot(t_1+\cdots+t_n)}|c, t_1, \ldots, t_n\rangle.$$

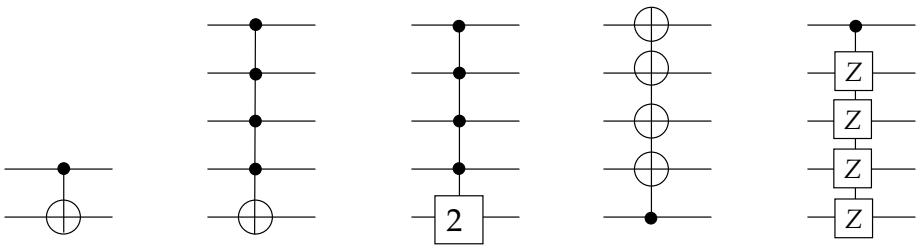

Figure 2.3: 2-qubit CNOT gate and other multi-qubit gates: Toffoli, parity, fan out and Z-fanout gate. The black dots represent the control qubits whose value does not change when acting on any standard basis state. The fanout gate is described in Section 2.2.2.

A quantum circuit is constructed out of layers. Each layer L is a tensor product of a certain fixed set of gates. A circuit is simply a (matrix) product of layers $L_1 L_2 \cdots L_d$. (Observe that the "last" layer L_d is actually the one that is applied directly to the inputs, and L_1 is the output layer.) The number of layers d is called the *depth* of C. The size of C is the number of gates in C. A circuit C over n qubits is then a unitary operator in $\mathcal{U}_n$. Clearly, C computes a unitary operator U *exactly* if for all computational basis states, $C|x_1, ..., x_n\rangle = U|x_1, ..., x_n\rangle$.

Often we will talk about a collection of circuits constructed using a *fixed finite set of gates*. For this purpose, we consider all gates of the same type but different fanin to be a single member of this set e.g. Toffoli gates with different fanin. Unless otherwise specified, all circuits should be understood to be elements of an infinite *family* of circuits

$\{C_n | n \geq 0\}$, where C_n is a quantum circuit for n input qubits and there is a fixed finite set of gates that can be used to construct C_n.

2.2.2 Fanout Gate

There is one other gate which will appear frequently in this thesis: the *fanout gate*. Fanout means creating multiple copies of a value; in classical circuit this is achieved by simply splitting a wire. The power of this trivial operation becomes apparent while designing quantum circuits because quantum computing does not allow creating copies of qubits[2].

To overcome this problem, a fanout gate, denote by F, can be used which unitarily copies a qubit only if it is in a standard basis state. Its action on standard basis states is given by

$$F|x, y_1, y_2, \ldots\rangle = |x, y_1 \oplus x, y_2 \oplus x, \ldots\rangle$$

This gate is called a fanout gate because $y_1, \ldots$ can be set to 0 to create multiple copies of $|x\rangle$. Note that a CNOT gate is same as a fanout gate on 2 qubits.

Unlike in classical circuits, fanout gates bring a lot of additional power to quantum circuits [HŠ03]. One of its most remarkable properties lies in its intimate relationship with the parity gate. There is no obvious *a priori* relation between these operators, and indeed we wouldn't expect there to be any on the basis of our experience with classical circuits. But as was observed by Moore [Moo99], F is conjugate to P via an $(n+1)$-fold tensor product of Hadamards applied to all the bits:

$$F = H^{\otimes(n+1)} P H^{\otimes(n+1)}$$

This is a consequence of the well-known fact when a CNOT gate is conjugated with Hadamards, the input and target qubit bits are flipped (see Figure 2.4).

It is immediate from this fact that, if we allow fanout for free, then unbounded-fanin quantum circuits can compute parity in constant depth and with polynomial size. Contrast this with the famous classical result of Furst, Saxe and Sipser [FSS84] that parity

[2]The basic idea of the *no-cloning theorem* is that creating copies of arbitrary quantum states is a non-unitary operation.

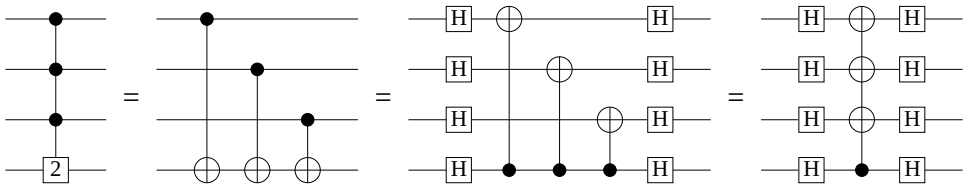

Figure 2.4: The parity and fanout gates are conjugates of each other by a layer of Hadamard gates.

cannot be computed by constant depth, polynomial size circuits using unbounded fanin AND and OR gates (besides NOT gates and implicitly using fanout).

2.2.3 Ancillæ

Sometimes circuits use additional inputs which are initialised to a fixed value. For classical circuits, it is understood that circuits always have access to as many values of 0 and 1 as needed. It is easy to generate a 0 from any input variable (e.g. $0 = x \oplus x$) and the circuit can generate any number of 0 and 1 from a single 0 using NOT gates and fanout. Thus workspace qubits are taken for granted while dealing with classical circuits.

However, due to the restriction of arbitrary fanout in quantum circuits, the number of workspace qubits is an important physical resource. These extra ancillary qubits, initialised to some fixed constant and used during the computation, are called *ancillæ*. Using ancillæ is considered expensive and we want to minimise their use and reuse them if possible. So a related question is what is the final state of the ancillæ. Ideally, we want to return the ancillæ to their starting state at the end of a circuit computation so that they can be reused if necessary in the future. A circuit is said to *cleanly compute* a function if the ancillæ needs to be initialised to a particular state and the ancillæ should be returned to that state at the end of the computation. In contrast to this, a circuit is said to *robustly compute* a function if it works for any initial state of the ancillæ. Their advantage is that several robust computing circuits can be composed together one after another because the ancillæ is not required to be in the initial state at the end of a robust computation.

Unless explicitly stated, all the circuits we consider here perform clean computation.

2.2.4 Universal quantum gates

A set of gates is said to be *universal* if they can be used to compute an arbitrary Boolean function (for classical circuits) or arbitrary operator (for quantum circuits). Some examples of universal sets of classical gates are {AND, NOT}, {NAND}.

We can similarly define universal quantum gates. There has been a lot of research on creating different universal sets of quantum gates. Practical considerations often decide which set of gates could be used in reality e.g. there might be hardware limitations or requiring the gates to be fault tolerant. Also, from a theoretical point of view, each universal set has its own characteristic property which maybe suitable for a particular kind of analysis.

Back to the basic question, can we implement any quantum circuit using a *fixed* set of gates? A fixed set of universal gates is required for practical reasons. By a result of Barenco et al. [BBC$^+$95], any unitary operator can be realised exactly with just CNOT and single-qubit gates. But an arbitrary unitary operator may involve complex numbers with arbitrary precision and it may not be possible to implement it using a fixed set of gates. On the other hand, it might be possible to approximate any operator with a fixed set of gates.

Definition 2.2.1 (Universal set) *A set of gates is said to be (effectively) universal if for any $n > 0$, any n-qubit unitary operator U and any error $\epsilon > 0$, there is an n-qubit unitary operator V that can be implemented using only gates from this set such that* $\max_{|\Psi\rangle} \|U|\Psi\rangle - V|\Psi\rangle\| < \epsilon$.

Indeed, the above paper also shows that Hadamard, $\pi/8$ and CNOT gate forms an universal set. Another set of universal gates that is often used contains the Hadamard, phase, CNOT and Toffoli gates. Unfortunately, the above definition does not say anything about the number of gates from the universal set needed to simulate any arbitrary operator. A simple counting argument shows that most n-qubit unitary operators require

exponentially many gates (exponential in n and $1/\epsilon$) from any universal set. An area of ongoing research is designing universal sets that can be used to efficiently simulate some particular group of operators.

If we instead want a simpler set of gates, then Shi [Shi02] and Aharonov [Aha03] showed that the Hadamard and Toffoli gates are universal for a relaxed notion of universality. They showed that if any n-qubit operator U can be implemented using, say, t single-qubit and two-qubit gates, then U can be approximated using $poly(n, t, 1/\epsilon)$ Hadamard and Toffoli gates. Recall that the Toffoli gate is essentially a classical gate but when used with the single qubit Hadamard gate, they can efficiently simulate any set of single and two qubit gates. This is interesting because the Gottesman-Knill theorem [NC00] showed that any quantum circuit using the CNOT, Hadamard and Pauli gates[3] can be simulated efficiently (polynomial steps in the number of qubits) on a classical computer.

2.3 Previous results

In this section we will briefly highlight some of the important results in quantum circuit complexity known today.

The quantum Fourier transform (QFT) is one of the most widely used unitary transformations in quantum circuits. It is one of the key components of many quantum algorithms, like Shor's [Sho97] quantum algorithm for factoring. An efficient implementation of the QFT will improve a wide variety of quantum circuits and algorithms. Its classical counterpart is known as the *discrete Fourier Transform* (DFT). The m-dimensional DFT maps $(a_0, \ldots, a_{m-1}) \in \mathcal{C}^m$ to $(b_0, \ldots, b_{m-1}) \in \mathcal{C}^m$ where,

$$b_x = \sum_{y=0}^{m-1} e^{(2\pi i/m)xy} a_y$$

The *fast Fourier transform* algorithm can compute the DFT in $O(m \log m)$ operations. The m-dimensional *quantum Fourier Transform* can be seen as a unitary operation performing

[3] This set of gates is called as the *Clifford group* and circuits constructed only using gates from this group are known as *Stabilizer circuits*.

the DFT on the amplitudes of a $\log m$-qubit state, mapping $\sum_{x=0}^{m-1} \alpha_x |x\rangle$ to $\sum_{x=0}^{m-1} \beta_x |x\rangle$ where,

$$\beta_x = \frac{1}{\sqrt{m}} \sum_{y=0}^{m-1} e^{(2\pi i/m)xy} \alpha_y$$

Høyer and Špalek showed that a constant depth circuit family is powerful enough to approximate the QFT.

Theorem 2.3.1 *[HŠ05] The QFT transformation can be approximated with polynomially small error by a quantum circuit of constant depth using CNOT and a fixed family of single-qubit gates.*

In the earlier section, we mentioned some surprising power of the fanout gate that does not have any parallel in the classical circuit model. Next we will describe an exponential improvement in the quantum model over the classical model, using the parity gate which is size and depth equivalent to the fanout gate.

Recall that a MOD_p gates computes a Boolean function that outputs 0 if and only if the sum of its binary inputs is $0 \pmod p$. The following theorem of Razborov/Smolensky characterises simulation of Mod_p gates by classical circuits.

Theorem 2.3.2 ([Raz87, Smo87]) *If p and q are distinct primes, a depth k circuit made of OR, AND, NOT and MOD_p gates cannot approximate MOD_q to within error $o(1)$ if it has size $2^{o(n^{1/2k})}$.*

Green et al. showed that unlike in the classical case, for quantum gates a MOD_2 gate is constant depth and polynomial size equivalent to a MOD_p gate for any p using single qubit and Toffoli gates [GHMP02]. This basically means all MOD gates are equivalent to each other for polynomial size circuits and furthermore, since parity (a MOD_2 gate) is equivalent to fanout, constant depth circuits with fanout can compute any MOD function. This is sharply in contrast with classical circuits (which trivially allow fanout) since they cannot compute any MOD function in constant depth using only AND, OR, NOT gates.

Another example where quantum circuits display radical improvements over their classical cousins are constant depth circuits made with bounded fanin gates. It is easy to

argue that such circuits cannot compute any useful function since no output can depend on all the inputs (we provide a technical proof of this later [Theorem 4.3.1]). However, Høyer and Špalek showed that if we also add unbounded fanout, then with high accuracy, one can compute *threshold* functions with bounded fanin gates in constant depth!

Theorem 2.3.3 *[HŠ05] Let $\{F_n\}$ be a family of operators computed by constant depth circuits built using single qubit gates, unbounded fanout gates and one of these cases:*

1. *constant fanin gates*
2. *unbounded fanin Toffoli gates*
3. *unbounded fanin quantum threshold gates*

Then there is a family of operators $\{G_n\}$ in either of the other cases that approximates $\{F_n\}$ with two-sided polynomially small error.

This theorem essentially says that fanout and single-qubit operations form a universal set of quantum gates. There are a number of immediate corollaries that follow from Theorem 2.3.3 when combined with some results on classical threshold circuits [SBKH93]. For example, iterated multiplication, division, and sorting of n integers can be done with polynomial-size circuits with threshold gates. Hence these operations can also be approximated by constant depth quantum circuits using bounded fanin gates and unbounded fanout gates.

2.4 Other models: Quantum decision tree

There are several other models of quantum computing that we will not discuss in detail in this thesis. However we will use some related results in Chapter 5. In this section, we introduce the quantum decision tree model and quantum query complexity.

The decision tree model is one of the simplest models of computing. Here the goal is to compute a Boolean function $f : \{0,1\}^n \to \{0,1\}$ given an input $x \in \{0,1\}^n$. In the

classical model, the algorithm is allowed to adaptively query any bit of the input and the cost of a computation is the number of such queries. We ignore other intermediate computations and so (for binary input strings) the algorithm looks like a binary tree with nodes labelled by x_i-s.

The quantum model is conceptually similar but defined in a way to ensure that all changes are unitary. The algorithm works on 3 registers I, B, W where I has $\log n$ qubits and is used to write a query, B has one qubit and is used to store the answer to a query and W is the workspace register with polynomially many qubits. The initial state of the algorithm is $|0\rangle_I \otimes |0\rangle_B \otimes |0\rangle_W$. The query steps are modelled as the application of the unitary operator O_x: $|i, b, w\rangle \rightarrow |i, b \oplus x_i, w\rangle$. The algorithm is allowed to perform intermediate computations between the queries in the form of unitary operators independent of the input. A k-query decision tree A is the unitary operator $A = U_k O U_{k-1} O \ldots U_1 O U_0$ and the output of the algorithm is the value obtained when the first qubit of $A|0, 0, 0\rangle$ is measured in any given basis.

Most of the currently known results in quantum complexity theory are for query complexity in the decision tree model [BW02]. The most famous result in this category is Grover's unordered search algorithm [Gro96]. The unordered search problem is as follows: given an arbitrary n-bit string x, the goal is find any i such that $x_i = 1$ or output -1 if no solution exists. The algorithm is allowed to query any x_i and the cost of the algorithm is the number of such queries. Grover constructed a bounded error quantum algorithm which takes $O(\sqrt{n/t})$ queries where t is the number of bits of x set to 1 (if there are no results i.e. $t = 0$, then the algorithm returns "no result" also in $O(\sqrt{n})$ queries). There are several extensions to the original algorithm. For example, if t is not known, then there is an algorithm which can still find a solution using same number of queries. There is also a counting version of the search algorithm; Brassard et. al. [BHT98] described a quantum algorithm which uses the techniques of Grover and Fourier transform and it can estimate t with very high probability in roughly $O(\sqrt{n})$ queries.

3 Universal Quantum Circuit

"Can physics be simulated by a universal computer? ... Now, what kind of physics are we going to imitate? ... But the physical world is quantum mechanical, and therefore the proper problem is the simulation of quantum physics..."

– Richard P. Feynman

In 1982, Richard Feynman raised the question of simulation of all kinds of physical processes by a computer [Fey82]. He argued that classical computers would not be able to simulate the quantum-mechanical processes; so he proposed the idea of a quantum model to capture the existing computations as well as probabilistic and quantum computations. There are simply too many kinds of computation happening in real world every moment, and logically there are ones that cannot even be solved[1]. Obviously, such a general machine, also known as a *universal machine*, would be very complicated.

In this thesis we are concerned with the quantum circuit model of computing; so we wonder if there can be such a machine in this model and if possible, how would it look? We define and construct efficient depth-universal and almost-size-universal quantum circuits. Such circuits can be viewed as general-purpose simulators for central quantum circuit classes and used to capture the computational power of the simulated class. This chapter is based on the paper

•[BFGH09] D. Bera, S. Fenner, F. Green and S. Homer. Efficient Universal Quantum

[1]For example, halting problem. It is logically impossible to always decide if any well structured program will ever halt [Tur36].

Circuits. Appeared in Proceedings of *The 15th International Computing and Combinatorics Conference*, 2009.

3.1 Introduction

Probably the most talked about universal machine is the universal Turing machine [Her95]. Alan Turing conceptualised the idea of an all-purpose machine that, given a meaningful but arbitrary sequence of instructions and any relevant data, could execute these instructions on these data. This machine, which he named *universal computing machine* [Tur36], is the basis of the stored-program architecture modern computers follow.

Most of the models we work with have equivalent computing power and can simulate each other; so, in a sense, the universal Turing machine ensures a universal machine for all the models. However, what is not clear is how complicated these machines are and how good are they at simulation. For example, it is a big challenge to come up with the smallest universal Turing machine; currently the smallest one has 2 states and 5 symbols and it is an open challenge to create one with 2 states and 3 symbols [Coo04].

Quantum computing is most naturally formulated using the quantum circuit model [Yao93]. Many quantum algorithms are expressed in terms of (uniform) special-purpose circuits which depend strongly on the problem being solved. However, the notion of a universal quantum computer is more naturally captured by quantum Turing machines [Deu85]. This being the case, it is desirable to have a notion of efficient universal quantum circuits in the spirit of universal quantum Turing machines.

Like resource-bounded universal Turing machines, efficiently constructed universal circuits characterise the hardest languages computed by circuits in a given circuit class. The existence of a universal circuit family for a complexity class defined by resource bounds (depth, size, gate width, etc.) provides an upper bound on the resources needed to compute any circuit in that class. More precisely, the specific, efficient construction of a universal circuit for a class of circuits with a fixed input length, yields a single circuit

which can be used to carry out the computation of every circuit in that class, basically a chip or processor for that class of circuits. The more efficient the construction of the universal circuit, the smaller and faster the processor for that class. For example, depth-universal circuits are desirable because they can simulate any circuit within a constant slow-down factor. Thus they are as time-efficient as possible.

Universal quantum circuits have been studied before in different contexts. Most of the research on universal quantum circuit classes deals with finding universal sets of gates which can be used to efficiently simulate any quantum computation ([NC00], [SR07]). Our goal is quite different; we want to create a circuit which, like a computer, takes as input both a program and data and runs the program on the data. Nielsen and Chuang considered a similar problem in [NC97]; however they did not focus on the efficiency of the universal circuit. They showed that it is not possible to have a generic universal circuit that works for all circuits of a certain input length. We avoid this problem by considering families of circuits with a certain depth or size and constructed from a fixed family of gates.

In the case of quantum circuits there are particular issues relating to the requirements that computations must be clean and reversible which come into play, and to an extent complicate the classical methods. Still, much of our motivation for this work originates with classical results due to Cook, Valiant, and others [CH85, Val76]. Cook and Hoover considered depth universality and described a depth-universal uniform circuit family for circuits of depth $\Omega(\log n)$. Valiant studied size universality and showed how to construct universal circuits of size $O(s \log s)$ to simulate any circuit of size s.

Definition 3.1.1 (Universal Quantum Circuits) *Fix $n > 0$ and let $\mathcal{C}$ be a collection of quantum circuits on n qubits. A quantum circuit U on $n + m$ qubits is* universal for $\mathcal{C}$ *if, for every circuit $C \in \mathcal{C}$, there is a string $x \in \{0,1\}^m$ (the* encoding*) such that for all strings $y \in \{0,1\}^n$ (the* data*),*

$$U(|y\rangle \otimes |x\rangle) = C|y\rangle \otimes |x\rangle.$$

The circuit collections we are interested in are usually defined by bounding various

parameters such as the size (number of gates), depth (number of layers of gates acting simultaneously on disjoint sets of qubits), or palette of allowed gates (e.g., Hadamard, $\pi/8$, CNOT).

As in the classical case, we also want our universal circuits to be *efficient* in various ways. For one, we restrict them to using the same gate family as the circuits they simulate. We may also want to restrict their size or the number m of qubits they use for the encoding.

For depth we show how to construct universal circuits whose depth is the same order as the circuits being simulated in Section 3.3. For size, there is a log factor blow-up in the universal circuits constructed here which is nearly optimal for polynomial size circuits (Section 3.4).

3.1.1 Other relevant work

The study of quantum circuit complexity was originated by Yao [Yao93]. The basic definitions and first results in this research area can be found in Nielsen and Chuang [NC00]. Most of the research on universal quantum circuit classes deals with finding small, natural, universal sets of gates which can be used in quantum circuits to efficiently simulate any quantum computation. Our problem and point of view here is quite different. We have the goal of constructing, for a natural class C of quantum circuits, a single family of quantum circuits which can efficiently simulate all circuits on the class C. In this chapter we consider classes C which have significant resource bounds (small or even constant depth, or fixed size) and ask that the corresponding universal circuit family have similar depth or size bounds.

Cook and Hoover [CH85] considered the problem of constructing general-purpose classical (Boolean) circuits using gates with fanin two. They asked whether, given n, c, d, there is a circuit U of size $c^{O(1)}$ and depth $O(d)$ that can simulate any n-input circuit of size c and depth d. Cook and Hoover constructed a depth-universal circuit for depth $\Omega(\log n)$ and polynomial size, but which takes as input a nonstandard encoding of the circuit,

and they also presented a circuit with depth $O(\log n \log\log n)$ to convert the standard encoding of the circuit to the required encoding.

Valiant looked at a similar problem—trying to minimise the size of the universal circuit [Val76]. He considered classical circuits built from fanin 2 gates (but with unbounded fanout) and embedded the circuit in a larger universal graph. Using switches at key vertices of the universal graph, any graph (circuit) can be embedded in it. He managed to create universal graphs for different types of circuits and showed how to construct a $O(c \log c)$-size and $O(c)$-depth universal circuit. He also showed that his constructions have size within a constant multiplicative factor of the information theoretic lower bound.

For quantum circuits, Nielsen and Chuang (in [NC97]) considered the problem of building generic universal circuits, or *programmable universal gate arrays* as they call them. Their universal circuits work on two quantum registers, a data register and a program register. They do not consider any size or depth bound on the circuits and show that simulating every possible unitary operation requires completely orthogonal programs in the program register. Since there are infinitely many possible unitary operations, any universal circuit would require an infinite number of qubits in the program register. This shows that it is not possible to have a generic universal circuit which works for all circuits of a certain input length. However they showed that it is possible to construct an extremely weak type of probabilistic universal circuit with size linear in the number of inputs to the simulated circuit.

Sousa and Ramos considered a similar problem of creating a universal quantum circuit to simulate any quantum gate [SR07]. They construct a basic building block which can be used to implement any single-qubit or CNOT gate on n qubits by switching certain gates on and off. They showed how to combine several of these building blocks to implement any n-qubit quantum gate.

3.2 Building Blocks: Quantum Gates

In order to construct efficient universal circuit families, it appears necessary to resort to the massive parallelism that fanout gates provide; note that the existing classical constructions make abundant use of fanout which is a very natural operation for classical circuits. It is an open question whether the same efficiency can be achieved without using fanout gates.

Our construction requires us to use controlled versions of the gates used in the simulated circuit. For most of the commonly used basis sets of gates (e.g., Toffoli gate, Hadamard gate, and phase gate S), the gates themselves are sufficient to construct their controlled versions (e.g., a controlled Hadamard gate can be constructed using a Toffoli gate and Hadamard and phase gates). Depth or size universality requires that the controlled versions of the gates should be constructable using the gates themselves within proper depth or size, as required.

Definition 3.2.1 (Closed under controlled operation) *A set of quantum gates $G = \{G_1, \ldots\}$ is said to be* closed under controlled operation *if for each $G_i \in G$, the controlled version of the gate C-$G_i|c\rangle|t\rangle \longrightarrow |c\rangle G_i^c|t\rangle$ can be implemented in constant depth and size using the gates in G. Here, $|c\rangle$ is a single qubit and G_i could be a single or a multi-qubit gate.*

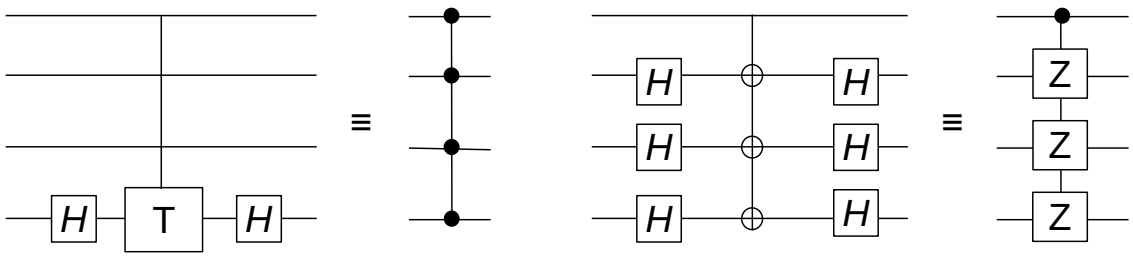

Figure 3.1: Equivalence of Toffoli gate and Z gate (left) and F gate and $zfanout$ gate (right)

A Z gate can be constructed easily (in constant depth and size) from a single unbounded Toffoli gate (and vice versa) by conjugating the target qubit of the unbounded

Toffoli gate with H gates (see Figure 3.1). Similarly, a Z_n gate can be constructed from a single F gate (and vice versa) in constant depth (although not constant size) by conjugating each target with H gates. So, in our depth-universal circuit construction, we can use either or both of these types of gates. Similarly for unbounded Toffoli versus Z gates. Z gates and Z-fanout gates are convenient to work with because they only change the phase, leaving the values of the qubits intact (they are represented by diagonal matrices in the computational basis). This allows us to use a trick due to Høyer and Špalek [HŠ03] and run all possible gates for a layer in parallel.

3.3 Depth-universal quantum circuits

Depth universal circuits are defined to have the same order of depth as the simulated circuits. We are particularly concerned with the depth of universal circuits. In computations modelled by circuits, circuit depth corresponds to time as we envision each layer of the circuit being carried out in one step of the computation by parallely applying all the gates in that layer. So, for example, depth-universal circuits are desirable because they can simulate any circuit within a constant slow-down factor. Thus they are as time-efficient as possible. If instead we want to minimise the number of gates that need to be applied, then size-universal circuits are a good candidate.

Definition 3.3.1 (Depth-Universal Quantum Circuits) *Fix a family $\mathcal{F}$ of unitary quantum gates. A family of quantum circuits $\{U_{n,d}\}_{n,d>0}$ is* depth-universal over $\mathcal{F}$ *if*

1. *$U_{n,d}$ is universal for n-qubit circuits with depth $\leq d$ using gates from $\mathcal{F}$,*
2. *$U_{n,d}$ only uses gates drawn from $\mathcal{F}$,*
3. *$U_{n,d}$ has depth $O(d)$, and*
4. *the number of encoding qubits of $U_{n,d}$ is polynomial in n and d.*

Cook and Hoover considered the problem of depth-universal classical circuit for circuits with only bounded-width gates [CH85]. As in their case, the following counting

argument shows that depth-universal quantum circuits, if they exist, must have depth $\Omega(\log n)$ and thus can only depth-efficiently simulate circuits with depth $\Omega(\log n)$. For simplicity, assume that we have a universal circuit U for depth-1 circuits on n qubits that uses CNOT gates *only*; we will use a communication type argument which can be trivially extended to any family of gates. Since any pair of the n qubits could potentially be connected with a CNOT gate, that pair must be connected somehow (indirectly perhaps) within the circuit U. Thus any data input qubit can potentially affect any of the other $n-1$ data output qubits. Since U only has constant-width gates, the number of qubits affected by any given data input increases by only a constant factor per layer, and so U must have $\Omega(\log n)$ layers. So for the rest of this section, assume that the circuits being simulated have depth $\Omega(\log n)$.

The construction of Cook and Hoover does not trivially extend to classical circuits with unbounded-width gates. However, in this section we For circuits with unbounded-width gates, in this section we prove that depth-universal circuits exist for unbounded-width circuits over each of the gate families

$$\begin{aligned} \mathcal{F} &= \{H, \pi/8\} \cup \{F \mid n \geq 1\}, \\ \mathcal{F}' &= \{H, \pi/8\} \cup \{F \mid n \geq 1\} \cup \{T \mid n \geq 1\}. \end{aligned}$$

We first give the proof for $\mathcal{F}$ then show how to modify it for $\mathcal{F}'$.

Theorem 3.3.2 *Depth-universal quantum circuits exist over $\mathcal{F}$ and over $\mathcal{F}'$. Such circuits use $O(n^2 d)$ qubits and can be built log-space uniformly in n and d.*

Our construction is quite different from the Cook and Hoover construction, mainly due to the limited ability to do fanout in quantum circuits. It seems impossible to perform efficient simulation without using fanout and so both the circuit families we consider consists of the fanout gate. Note that the results for the two circuit families are independent, because it is not known whether n-qubit Toffoli gates can be implemented exactly in constant depth using single-qubit gates and fanout gates, although they can be approximated this way [HŠ03].

The depth-universal circuit U we construct simulates the input circuit C layer by layer, where a layer consists of the collection of all its gates at a fixed depth. C is encoded in a slightly altered form, however. First, all the fanout gates in C are replaced with Z-fanout gates on the same qubits with H gates conjugating the targets. At worst, this may roughly double the depth of C (adjacent H gates cancel). Each layer of the resulting circuit is then separated into three adjacent layers: the first having only the H gates of the original layer, the second only the $\pi/8$ gates, and the third only the Z-fanout gates. U then simulates each layer of the modified C by a constant number of its own layers. We describe next how these layers are constructed.

Simulating single-qubit gates. The circuit to simulate an n-qubit layer of single-qubit gates of type G, say, consists of a layer of controlled-G gates where the control qubits are fed from the encoding and the target qubits are the data qubits. Figure 3.2 shows a layer of G gates, where $G \in \{H, \pi/8\}$, controlled using H, S, $\pi/8$, CNOT, and Toffoli gates To simulate G gates on qubits $i_1, \ldots, i_k$, say, set $c_{i_1}, \ldots, c_{i_k}$ to 1 and the rest of the c-qubits to 0.

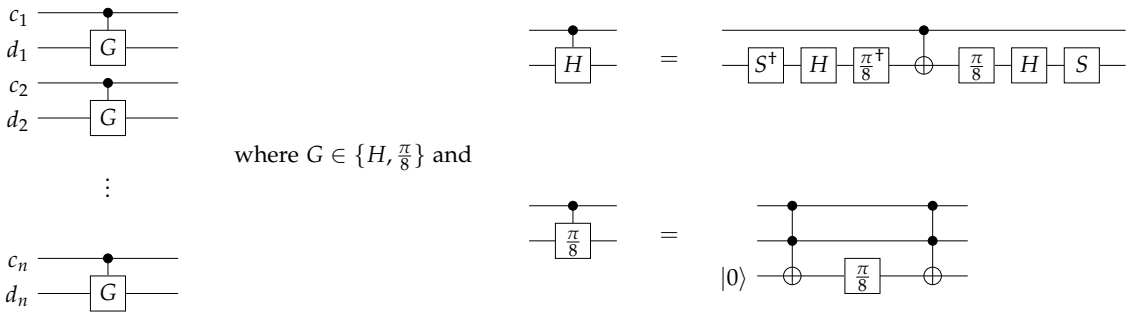

Figure 3.2: Simulating a layer of single-qubit G gates with controlled G gates. The ancilla in the implementation of the controlled $\pi/8$ gate is assumed part of the encoding. The ancilla is reset to 0 at the end and hence can be reused for implementing all $\pi/8$ layers.

The gates used in Figure 3.2 that are not in $\{H, \pi/8\}$ can be actually implemented using the gates from this set. For example, we can implement S by $\frac{\pi}{8}^2$, and since $\frac{\pi}{8}^8 = I$, we can implement $S^\dagger = \frac{\pi}{8}^6$ and $\frac{\pi}{8}^\dagger = \frac{\pi}{8}^7$.

Simulating Z-fanout gates. The circuit to simulate a Z-fanout layer is shown in Figure 3.3. The top n qubits are the original data qubits. The rest are ancilla qubits. All the

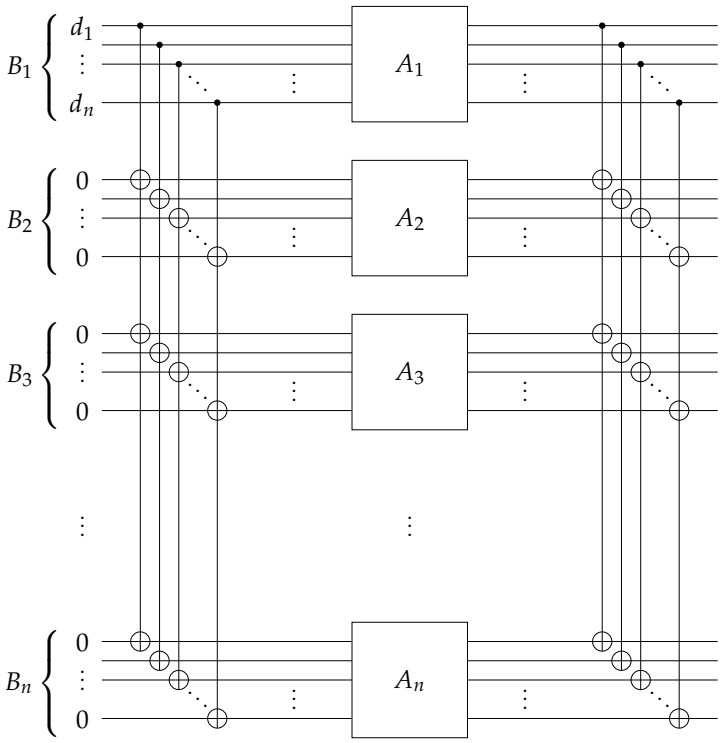

Figure 3.3: Simulating a layer of Z-fanout gates.

qubits are arranged in n blocks $A_1, \ldots, A_n$ of n qubits per block. The qubits in block A_i are labelled $b_{i1}, \ldots, b_{in}$.

Each A_i subcircuit looks like Figure 3.4. The qubits $c_{i1}, \ldots, c_{in}$ are encoding qubits. The large gate between the two columns of Toffoli gates is a Z-fanout gate with its control on the ith ancilla (corresponding to b_{ii} and c_{ii}) and targets on all the other ancillæ.

Here is the state evolution from $|\vec{d}\rangle = |d_1 \cdots d_n\rangle$, suppressing the c_{ij} qubits and ancillæ internal to the A_i subcircuits in the ket labels. Note that after the first layer of fanout gates,

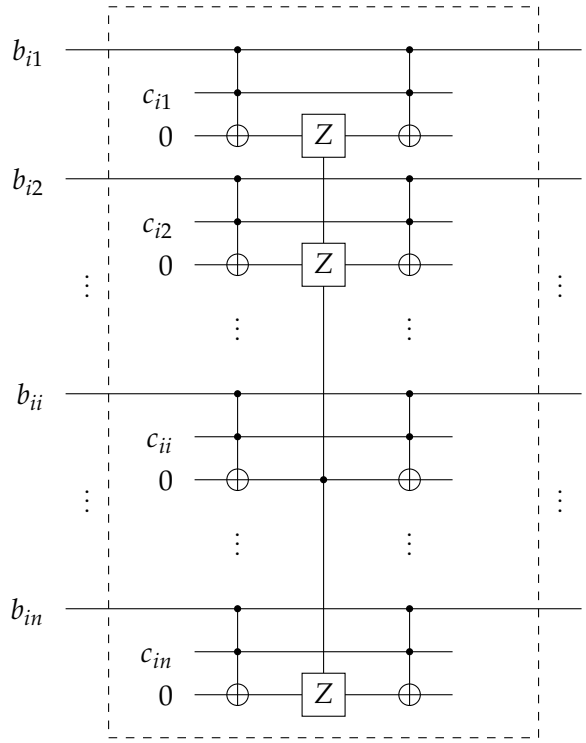

Figure 3.4: Subcircuit A_i in the simulation of Z-fanout gates.

each qubit b_{ij} carries the value d_j.

$$\begin{aligned} |\vec{d}, \vec{0}, \ldots, \vec{0}\rangle &\mapsto |\vec{d}, \vec{d}, \ldots, \vec{d}\rangle \\ &\mapsto (-1)^{\sum_i d_i c_{ii} \left(\sum_{j \neq i} d_j c_{ij}\right)} |\vec{d}, \vec{d}, \ldots, \vec{d}\rangle \\ &\mapsto (-1)^{\sum_i d_i c_{ii} \left(\sum_{j \neq i} d_j c_{ij}\right)} |\vec{d}, \vec{0}, \ldots, \vec{0}\rangle. \end{aligned}$$

To simulate some Z-fanout gate G of C whose control is on the ith qubit, say, we do this in block B_i by setting c_{ii} to 1 and setting c_{ij} to 1 for every j where the jth qubit is a target of G. All the other c-qubits in B_i are set to 0. We can do this in separate blocks for multiple Z-fanout gates on the same layer, because no two gates can share the same control qubit. Any c-qubits in unused blocks are set to 0.

Simulating unbounded Toffoli gates. We can modify the construction above to accommodate unbounded Toffoli gates (gate family $\mathcal{F}'$), or equivalently Z gates, by breaking each layer of C into four adjacent layers, the first three being as before, and the fourth

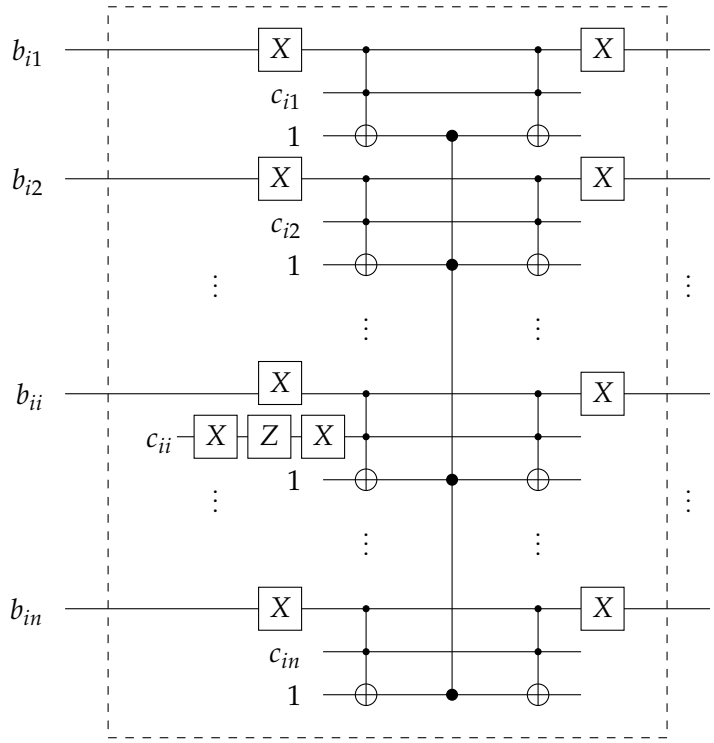

Figure 3.5: Subcircuit A_i for a layer of Z gates.

containing only Z gates. The top-level circuit to simulate a layer of Z gates looks just as before (Figure 3.3), except now each A_i subcircuit looks a bit different and is shown in Figure 3.5, where the central gate is a Z gate connecting the ancillæ.

As before, the qubits $c_{i1}, \ldots, c_{in}$ are encoding qubits. The XZX gates on c_{ii} multiply the overall phase by $(-1)^{\overline{c_{ii}}}$. When the Z gate of A_i is applied, its jth contact point is in the state $\overline{\overline{b_{ij}} c_{ij}}$. Note that $\overline{\overline{b_{ij}} c_{ij}} = b_{ij}$ if $c_{ij} = 1$ and 1 otherwise. The Z gate then multiplies the overall phase by $(-1)^{\prod_j (\overline{\overline{b_{ij}} c_{ij}})} = (-1)^{\prod_{j:c_{ij}=1} b_{ij}}$. The state thus evolves as given below:

$$\begin{aligned} |\vec{d}, \vec{0}, \ldots, \vec{0}\rangle &\mapsto |\vec{d}, \vec{d}, \ldots, \vec{d}\rangle \\ &\mapsto (-1)^{\sum_i \overline{c_{ii}} + \prod_{j:c_{ij}=1} b_{ij}} |\vec{d}, \vec{d}, \ldots, \vec{d}\rangle \\ &\mapsto (-1)^{\sum_i \overline{c_{ii}} + \prod_{j:c_{ij}=1} b_{ij}} |\vec{d}, \vec{0}, \ldots, \vec{0}\rangle \end{aligned}$$

To simulate some Z gate G of C whose first qubit is i, say, we do this in block B_i by setting c_{ii} to 1 and setting c_{ij} to 1 for every j where the jth qubit is part of G. All the other c-qubits in B_i are set to 0. As before, we can do this in separate blocks for multiple gates on the

same layer, because no two gates can share the same first qubit. Any c-qubits in unused blocks are set to 0, and it is easy to check that this makes the block have no net effect.

3.4 Size-universal quantum circuits

Similar to a depth-universal circuit, a *size-universal circuit* is a universal circuit whose number of gates is of the same order as the number of gates in the circuit it is simulating. Formally,

Definition 3.4.1 *A family* $\{U_{n,c}\}$ *of universal circuits for n-qubit circuits of size* $\leq c$ *is* size-universal *if* $SIZE(U_{n,c}) = O(c)$.

A simple counting argument shows that it is not possible to obtain a completely size-universal circuit for fanin-2 circuits. Consider all circuits with c fanin-2 gates where one input of each gate is connected the first qubit. There are $(n-1)^c$ possible circuits. Then consider similar circuits where there is no gate with input as the first qubit and one input of each gate is connected to the second qubit. Continuing this recursive counting argument, we obtain that the number of possible fanin-2 circuits is $\Omega((n-1)^{c+1})$. Since the universal circuit must be connected to all the encoding bits, it must have $\Omega(c \log n)$ gates.

We use Valiant's idea of universal graphs [Val76] to construct a universal family of fanin-2 circuits that are very close to the aforementioned lower bound. As before, we would like to simulate C by using the same set of gates used in C. Our construction works for any circuit using unbounded Toffoli gates and any set of single-qubit and 2-qubit gates closed under the controlled operation.

First we will define a universal directed acyclic graph with n special vertices (called *poles*) in which we can embed any circuit with n gates (counting the inputs as gates too). The embedding will map the wires in the circuit to paths in the graph.

Definition 3.4.2 (Edge-embedding [Val76]) *An* edge-embedding ρ *of* $G = (V, E)$ *into* $G' =$

(V', E') maps V one-to-one to V' and maps each edge $(i, j) \in E$ to a directed path $\rho(i) \rightsquigarrow \rho(j)$ in G' such that distinct edges are mapped to edge-disjoint paths.

The graph of any fanin-2 circuit of size n can be represented as a directed acyclic graph with vertices $\{1, \ldots, n\}$ such that there is no edge from j to i for $i < j$ and each vertex has fanin and fanout 2. Let $\Gamma_2(n)$ be the set of all such graphs.

Definition 3.4.3 (Edge-universal graph [Val76]) *A graph G' is* edge-universal *for $\Gamma_2(n)$ if it has distinct poles $p_1, \ldots, p_n$ such that any graph $G \in \Gamma_2(n)$ can be edge-embedded into G' where each vertex $i \in G$ is mapped to vertex $\rho(i) = p_i \in G'$.*

Then, Valiant shows how to construct a universal graph.

Theorem 3.4.4 ([Val76]) *There is a constant k such that for all n there exists an acyclic graph G' that is edge-universal for $\Gamma_2(n)$, and G' has $kn \lg n$ vertices, each vertex having fanin and fanout 2.*

It is fairly easy to construct a universal circuit using the universal graph. In fact, the universal circuit for circuits with n inputs and c gates will be any edge-universal graph for $\Gamma_2(n + c)$.

Consider any such edge-universal graph G'. Then G' has $c' = k(n + c) \log(n + c)$ vertices for some k. These c' vertices include fixed poles $p_1, \ldots, p_n, p_{n+1}, \ldots, p_{n+c}$ and non-pole vertices. Create a quantum circuit C' with c' gates (including the inputs and outputs) where G' describes how the gates connect to each other. For each of the vertices $p_1, \ldots, p_n$ of G', remove their incoming edges and replace the vertices by the input as shown in Figure 3.6. Replace each of the vertices $p_{n+1}, \ldots, p_{n+c}$ with a subcircuit that applies any of the single- or 2-qubit gates on the inputs, where the gate to apply is controlled by the encoding. For example, Figure 3.8 shows the subcircuit at a pole vertex in a universal circuit simulating CNOT and H gates; refer to figure 3.2 for the construction of the controlled-H gate.

For a non-pole vertex, replace it with a subcircuit that swaps the incoming and outgoing wires (i.e., first input is connected to second output and second input is connected to first output) or directly connects them (i.e., first input is connected to first output and similarly for the second input). Again, the subcircuit is controlled by the encoding which controls whether to swap or directly connect (see Figure 3.7). The edge disjointness property guarantees that wires in the embedded circuit are mapped to paths in C' which can share a vertex but cannot share any edge.

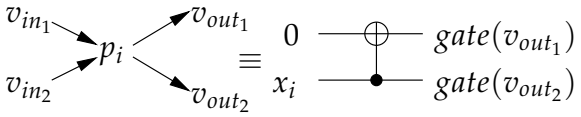

Figure 3.6: The gate for a pole vertex p_i is mapped to input x_i.

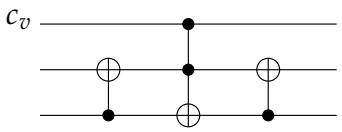

Figure 3.7: The gates at a non-pole vertex v. The encoding bit c_v specifies if first output qubit should be mapped to first input or second input qubit and similarly for second output qubit.

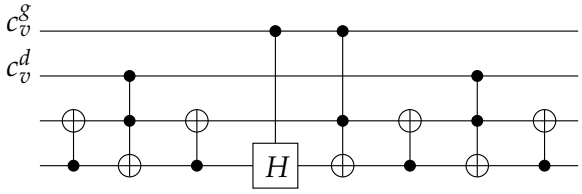

Figure 3.8: Example of the gates at a pole vertex v simulating a circuit with CNOT and H gates. The encoding bits c_v^g specify which kind of gate is at vertex v, and the c_v^d specify which qubit the gate acts on (for H gate) or which is the control qubit (for CNOT gate).

To simulate any fanin-2 circuit C with c gates acting on n qubits, construct the edge-universal graph G' for $\Gamma_2(n+c)$. Embed the graph of C into G' such that the input nodes of C are mapped to the poles $p_1, \ldots, p_n$ in G'. In the encoding, set c_v^g to denote the type of g. If g is a single-qubit gate, set c_v^d to denote which of the two input qubits is connected to g; if g is a 2-qubit gate, set c_v^d to denote the control qubit. Consider all

the non-pole vertices v. By the property of the edge embedding, every wire in the circuit can be mapped to an *edge disjoint* path in G. Consider each wire in and its corresponding path in G. Since the paths are all edge disjoint, the wire connections can be specified by specifying how the 2-output wire are connected to the 2 input wires. Set c_v to denote whether the wires should be directly connected or swapped.

The size of the encoding is $(n+c)(\log|\text{gates}|+1)+(|\Gamma_2(n+c)|-(n+c))$. For polynomial size circuits, it is $O(c\log c)$ and the construction gives us a universal circuit with a logarithmic blow-up in size.

Theorem 3.4.5 *There is a constant k and a family of universal circuits $U_{n,c}$ that can simulate every circuit with c gates acting on n qubits such that $SIZE(U_{n,c}) = O((n+c)\log(n+c))$.*

We can use a similar idea for circuits with unbounded fanin. First we decompose the unbounded fanin gates using bounded fanin gates (fanin 2 in this case). This is doable for most of the common unbounded fanin gates. For example, an unbounded Toffoli gate of size f can be constructed using $\Theta(f)$ successive Toffoli gates of size 3, which can in turn be implemented using Hadamard, phase, $\pi/8$ and CNOT gates [NC00]. So any circuit of size c consisting of Hadamard, $\pi/8$ and unbounded Toffoli gates can be transformed into an equivalent circuit with size at most $O(cn)$ consisting of these single-qubit gates and CNOT gates. The rest of the construction follows as before.

Corollary 3.4.6 *There is a family of universal circuits $U_{n,c}$ that can simulate quantum circuits of size c on n qubits and consisting of Hadamard, $\pi/8$, and unbounded Toffoli gates such that $SIZE(U_{n,c}) = O(nc\log(nc))$.*

3.5 Other results

Depth-universal classical circuits. The techniques of Section 3.3 can be easily adapted to build depth-universal circuits for a variety of classical (Boolean) circuit classes with unbounded gates, e.g., AC, ACC, and TC circuits. The key reason is that these big gates

are all "self-similar" in the sense that fixing some of the inputs can yield a smaller gate of the same type.

Circuit encoding. We have been mostly concerned with the actual simulation of a quantum circuit C by the universal circuit U. It is possible, however, to hide some complexity of the simulation in U's description of C itself. Usually, the description of a classical circuit describes the underlying graph of the circuit and specifies the gates at each vertex. We can similarly describe a quantum circuit by its graph structure. The description is extremely compact with size proportional to the size of the circuit. For example, the standard encoding of fanin 2 graphs would be a list of 4-tuples, one for each gate, (v, g, in_1, in_2). Here v is the gate number and g is the type of the gate. The 1st input of v is connected to the in_1 qubit and the 2nd input of v is connected to the in_2 qubit.

However, we used a description that is more natural for quantum circuits and especially suitable for simulation. The description stores the grid structure of the circuit; the rows of the grid correspond to the qubits, and the columns correspond to the different layers of the circuit. We first convert the circuit to have only one kind of gate at each layer; also we fix some ordering of the fixed finite set of gates and assume that layers in the circuit always consist only of one of the finite number of different gates, and the gates in each layer of gates always occur in the same order. (There can be layers with no gates.) This can all be done with a constant increase in depth as the number of different gates types is finite.

Then, for a layer of single qubit gates of type, say G, the grid encoding stores n Boolean values $(c_1, \cdots, c_n)$ where c_i denotes whether there is a G gate in that layer and on ith qubit. For a layer of controlled-Z gates, the encoding contains n^2 Boolean values $(c_{11}, c_{12}, \cdots, c_{nn})$ where c_{ii} denotes if there is a controlled-Z gate in that layer with the control on the ith qubit and c_{ij} denotes if there is a controlled-Z gate in that layer with control on the ith qubit and target on the jth qubit. For a layer of generalised Z gates, it contains n^2 Boolean values $(c_{11}, c_{12}, \cdots, c_{nn})$ where c_{ii} denotes if there is a generalised

Z gate in that layer with the first qubit on the ith qubit and c_{ij} denotes if there is a generalised Z gate in that layer on qubit j whose first qubit is on the ith qubit. This description is not unique for any given circuit and its size is $O(nd)$, where n is the number of qubits and d is the depth of the circuit. However, a graph-based description can be easily converted to this grid-based description in polynomial time.

3.6 Summary

In this chapter we showed how to construct universal circuit families with the desired property that they require (roughly) no more resource (depth or size) than the circuit they are simulating. Designing such quantum circuits is a challenge because the simulated circuits could perform operations involving one or both of the phase and states of the qubits which it is not immediately clear how to simulate. Furthermore, the universal circuit in itself is constrained by clean computation, lack of arbitrary copying of qubits and other such requirements of a quantum circuit.

Still, a number of natural, interesting open problems remain.

Fanout gates are used in our construction of a depth-universal circuit family. Is the fanout gate necessary in our construction? We believe it is. In fact, we do not know how to simulate depth-d circuits over $\{H, \pi/8, \text{CNOT}\}$ universally in depth $O(d)$ without using fanout gates, even assuming that the circuits being simulated have depth $\Omega(\log n)$. The shallowest universal circuits with bounded-width gates we know of have a $\lg n$ blow-up factor in the depth, just by replacing the fanout gates with log-depth circuits of CNOT gates.

Our results apply to circuits with very specific gate sets. How much can these gate sets be generalised? Are similar results possible for any countable set of gates containing Hadamard, unbounded Toffoli, and fanout gates?

We showed how to construct a universal circuit with a logarithmic blow-up in size. The construction is within a constant factor of the minimum possible size for polynomial-

size, bounded-fanin circuits. However for constant-size circuits, we believe the lower bound can be tightened to match the proven upper bound. For unbounded-fanin circuits, we construct a universal circuit with size $O(nc \log nc)$ which is significantly larger than the bounded fanin lower bound of $\Omega(c \log n)$. We think that a better lower bound is possible for the unbounded-fanin case.

4 Quantum Circuit Lower Bound

"I refuse to answer that question on the grounds that I don't know the answer."
– Zaphod Beeblebrox (H2G2 *by* Douglas Adams)

Our intelligence is of course no match for that of a Galactic ex-President, but knowledge about what we cannot do often carries significant insight about what we can. Computer science and logic, rather mathematics in general, is noteworthy for its openness towards results revealing its limitations. In the previous chapter we saw evidence that the quantum circuit model is powerful enough to let us create a single circuit with limited depth (or size) that can simulate all circuits of the same depth (or size). There are many examples of quantum circuits, even structurally very simple ones with constant or logarithmic depth, that possess more power than similarly resource bounded classical circuits. But can such circuits compute arbitrarily hard functions? In this chapter, we investigate their limitations. Our main contribution here is a new technique for proving lower bounds of quantum circuits. This chapter is based on the following report:

•[Ber08] D. Bera. A New Lower Bound Technique for Quantum Circuits without Ancillæ. Boston University, Computer Science technical report 2008-15 (2008).

4.1 Introduction

There are many models of computing used in computer science, e.g. Turing machines, circuits, decision trees to name a few. Each of the models have their own advantages. The

circuit model is particularly useful for proving lower bounds. Unlike a Turing machine, a circuit computing a function is easy to write down and is closely linked to the algebraic and combinatorial properties of the function. This makes it tempting to try to bound the power of the circuit, when limited to a certain depth or size or both.

There is a considerable interest in low depth quantum circuits. The recent developments in physical realisation of qubits and quantum gates suggest that the early quantum circuits will be of limited size and depth, if built using the classical like model of a network of gates. For example it has been shown that the quantum Fourier transform can be implemented in low depth and size [CW00]. Note as well that the quantum Fourier transform plays an important role in providing quantum speedups for many problems.

Many of the currently known upper bounds in quantum circuit complexity require the *fanout gate* [HŠ03], which interestingly enough, was found to be depth-3 equivalent to the parity gate [Moo99]. On one hand this makes the upper bounds stronger since a fanout gate seems to be something that could be physically implementable [Fen03b]. On the other hand, this begs the question whether something as simple as the fanout gate could be implemented in constant depth using the typical set of universal gates. Since fanout is equivalent to parity, this is a quantum analogue of the classical "PARITY $\in AC^0$" question [Bea94]. We address a limited form of this question in this chapter.

Apart from a resolution of the question of fanout gates with constant depth circuits, we are also interested in lower bound techniques for quantum circuits. Today there are several lower bound techniques for classical circuits e.g. using algebraic representation of the gates, randomly fixing inputs to the gates or information theoretic arguments [Pud93]. However, similar to other models for quantum computation, even to achieve more modest results than what is known for classical circuits, new techniques are needed. Also, different techniques reveal important insight about the structure which enriches our understanding of the model. To this effect, we describe a new technique to analyse quantum circuits without any ancillæ made of Toffoli gates and any single qubit gates.

4.2 Motivation and related work

In this chapter we solely focus on quantum circuits made with arbitrary fanin Toffoli gates and any set of single qubit gates. Our motivation for choosing this set of gates is two-fold. First, there are several (effectively) universal set of quantum gates consisting of a particular set of single qubit gates along with the Toffoli gate i.e. any quantum gate can be approximated arbitrarily closely using this set of gates [NC00]. Furthermore, any constant fanin gate can be effectively replaced by a constant depth and constant size circuit made of single qubit and CNOT gates; since a CNOT gate is the same as a Toffoli gate for 2 qubits, this means the forthcoming results also hold true for quantum circuits with arbitrary fanin Toffoli gates and any set of *bounded fanin* gates.

The second motivation is in tune with our quest in understanding what the quantum model brings to the plate. The Toffoli gate is very similar to the classical AND gate. So structurally the only difference is the presence of the single qubit quantum gates. Informally, the single qubit gates manipulate and share information locally (i.e. with a constant number of qubits at any time) and the unbounded classical gates act globally to share these changes.

In the analysis of such circuits we find it necessary to grapple with another likely limitation of real quantum computers. It is evident that they will be limited not only in their run-time duration but also by the number of qubits used in the computation, due to the difficulty in controlling the interactions of multiple qubits. It will be necessary to identify computations which use as few ancillæ as possible. Thus we consider here the number of ancillæ used by a circuit as an additional computational resource, and in addition to the above requirements, we also require that our circuits use no ancilla. Currently it is not known if such circuits are necessarily weaker than circuits with ancillæ.

Earlier in Chapter 2, we defined that a cleanly computing circuit returns the ancillæ to their initial states. In this chapter, we use "clean computation" in a more general sense. For quantum circuits that compute a Boolean function, usually there is a fixed

qubit whose measurement output is considered as the output value of the circuit. Such a circuit is said to cleanly compute that function, if *all* the qubits, excluding the measured qubit, are returned to their initial states. This basically extends the requirement from ancillæ to every qubit that they should be reusable if possible.

Unlike some other models, we do not require clean computation. This is okay since we are interested in lower bounds in this chapter; a lower bound for parity when clean computation is not required also gives a lower bound for parity when clean computation is required. We hope that these simplifications will lead to better understanding of the major underlying differences like entanglement and superposition.

Our technique is based on the usual observation about AND gates: even though an AND gate might act on a large number of inputs, for most input assignments the result is simply zero. Different manifestations of this observation have led to a number of techniques in the past [RW04].

The *random restriction* approach [Has86] notes that setting only one input of an AND gate to 0 is sufficient to replace the gate by a constant. It then uses this idea to randomly set several of the input variables and *kill* most of the gates of the circuit.

The algebraic techniques [Gre99, Bei93] represent the inputs as variables and the gates as polynomial functions on these variables. The main idea here revolves around approximating AND or OR by a low-degree polynomial function. The polynomials are then composed gate-by-gate and a low-degree polynomial is obtained for the whole circuit.

The quantum formula model is a restricted version of the quantum circuit model. Quantum formulæ are defined like Boolean formulæ, i.e. there is at most one path from any input to the output. For a quantum formula, this means that only one of the output qubits of any gate can be connected to the output qubit of the circuit. Rowchoudhury and Vatan looked at the quantum formula lower bound problem and gave an almost-quadratic lower bound of $\Omega(n^2/\log^2 n)$ for the problem of element distinctness on n inputs [RV01].

For quantum circuits, Fang et al. in [FFG+06] looked at the same problem as ours: a lower bound for parity using unbounded fanin Toffoli and other bounded fanin gates.

Their approach is similar to the *random restriction* method; they eliminate Toffoli gates by carefully setting input variables (however, they set the input selectively, unlike randomly as in the random restriction method). For a comprehensive understanding of the different lower bound techniques for quantum circuits, we briefly survey their method in Section 4.4.1. We then describe our approach in detail in Section 4.4.2.

4.3 Bounded fanin circuits

An oft used technique in circuit complexity is the communication argument that the output gate of a circuit must be ultimately connected to all the inputs that its corresponding function non-trivially depends on. This is also true for quantum circuits but the proof is not so obvious since the intermediate quantum states might be entangled. An application of this claim is that any quantum circuit with bounded fanin gates require depth at least $\log n$ to compute any function that is dependent on all input qubits e.g. parity, Toffoli or fanout. This was first proved in [FFG+06]; here we present a slightly different argument that will be helpful in understanding the later results in this chapter.

Theorem 4.3.1 *For any circuit from a family of quantum circuits made of bounded fanin gates and of depth d, the output of measurement of a single qubit can depend on at most $2^{O(d)}$ inputs.*

Proof. Consider a circuit of depth d with n inputs qubits and a ancillæ. For simplicity assume it is made of single qubit and 2-qubit gates. Also assume at the end of the circuit, the measurement of the first qubit in some fixed basis is output as the circuit output (the theorem is true as long as any constant number of output qubits are measured).

Since the gates are of fanin at most 2, at depth d, the state of the first qubit during measurement should depend at most on a fixed set of 2^d input qubits. However, it is possible that the quantum gates may have entangled all the input qubits. We claim that the measurement output of the first qubit does not depend on any such entanglement. Here is an explicit proof of this fact.

For any quantum circuit with single and 2-qubit gates, there is an equivalent directed acyclic graph where single qubit gates correspond to vertices with fanin and fanout 1 and 2-qubit gates correspond to vertices with fanin and fanout 2. Let the vertex for the last gate on the first qubit be denoted by u (the measurement qubit). Then the vertices can be partitioned into two disjoint subsets U and V such that U is the smallest subset that consists of u and other vertices from which u can be reached; observe that there is no edge from V to U and $|U| \leq 2^d$.

Since there is no edge from V to U, this means in the circuit, the gates in U do not depend on any gates from V. So, the circuit of depth d can be transformed to an equivalent depth $2d$ circuit where the gates in V are *pushed* past layer d i.e. if V has a gate g in layer i, then the new circuit has g in layer $d + i$. The transformed circuit then has all gates from U in the first d layers working only on 2^d qubits, followed by d more layers consisting on gates from V working on all but the first qubit (since U contains u and u is the last gate on first qubit, there is no outgoing edge from u). Formally, $C = (I^{(1)} \otimes V^{(n+a-1)})(U^{(2^d)} \otimes I^{(n+a-2^d)}))$ (the superscript denotes the size of the subspace the operator).

Since later gates cannot affect the measurement of the output of u, and the gates up to and including u depend only on 2^d gates, the measurement of the output of u can depend on at most 2^d inputs. □

Note that this argument works with any numbers of ancillæ. A careful analysis of the dependency of the output qubit gives us the exact depth required to compute parity and fanout.

Corollary 4.3.2 *[FFG+06, Th. 3.2] Any quantum circuit on n inputs and any number of ancillæ must have depth at least $\log n$ to exactly compute parity and depth at least $\log n - 2$ to exactly compute fanout.*

4.4 Unbounded fanin circuits

Next we move on to circuits with no ancillæ and composed of unbounded Toffoli gates and any set of single qubit gates. Both the techniques covered here are inspired by observations for classical circuits. We first survey an earlier approach by Fang et al. in [FFG+06].

4.4.1 Earlier technique

The crucial observation behind this technique is that the output of a classical AND or OR gate can be fixed by fixing only one input i.e. the gate can be effectively *killed* by fixing one input. Thus, if we consider a constant depth classical circuit made of NOT gates and unbounded fanin AND and OR gates but *without* any fanout, then its output can be fixed by fixing a few of its inputs.

A similar argument works for quantum circuits but not surprisingly, the method becomes very complicated since the intermediate states in the circuit can be a superposition of the basis states and also entanglement can create undesired dependencies between qubits. The technique employs two tricks to its advantage. First, it assumes that the circuit is composed of unbounded Z gates instead of Toffoli gates. Z gates, as described earlier, flip the phase of a basis state if all the qubits are 1; recall that a Z gates is constant depth and constant size equivalent to a Toffoli gate, so this replacement does not change the order of the size or depth of the circuit. Secondly, say a circuit C of depth d is composed of levels $L_1, \ldots, L_d$ such that $C|\Psi\rangle = L_1 \cdot L_2 \ldots L_d|\Psi\rangle$ and each L_i is a tensor product of Z gates and single qubit gates. Without loss of generality, assume that the nth qubit is measured as the output of C. Then, we can start killing Z gates backwards from level L_d to L_1: fixing the state of a set of qubits in each level as we go backward so that the output qubit is always in the state $|0\rangle$. It can be shown that the set of qubits whose states needs to be fixed in level L_i consists of at most 2^i qubits. Therefore, for the full circuit of d levels, there exists a state $|\Psi_{2^d}\rangle$ involving only 2^d qubits such that no matter

what the other $n - 2^d$ qubits are set to, after C is applied, measurement of the nth qubit will always output 0. With a careful analysis, [FFG+06] showed the following:

Theorem 4.4.1 (Lemma 4.2 [FFG+06]) *Consider any quantum circuit C with depth d, with n inputs and 0 ancillæ, and whose output is determined by measuring a particular qubit after C is applied to the input. If C is built is unbounded Toffoli gates and bounded width gates, then the output qubit can be fixed to $|0\rangle$ by fixing the state of $2^{d/2}$ of the n input qubits.*

They were able to extend their technique to include sublinear number of ancillæ, specifically the above result hold if C has a ancillæ as long as $n > (a+1)2^{d/2}$.

4.4.2 Our technique

Our approach is inspired by one of the algebraic approaches by Smolensky [Smo87] where classical AND gates are approximated as:

$$\forall \delta, \exists \text{polynomial } \hat{A}(X), \Pr_{X \in \mathcal{D}^n} \left[AND(X) = \hat{A}(X) \right] > 1 - \delta$$

where $\mathcal{D}^n$ is any distribution on n-bit strings and $\hat{A}(x)$ is a carefully constructed low-degree polynomial with degree dependent on δ. The crucial observation is that, for quantum circuits *without any ancillæ*, the reversibility property of the circuit ensures that the output distribution after any gate is same as the input distribution. Thus, if the input distribution is chosen uniformly at random, we can approximate AND by the simple degree-0 polynomial $\hat{A}(X) = 0$; note that though inspired by the technique of Smolensky, we do not need the whole machinery of his technique for our case. Since the states of a quantum circuit can be a superposition of basis states, instead of actually bounding the probability of approximating the AND gate, we bound the distance between the states obtained by applying the Toffoli gate and by applying the approximation. Thus we can approximate the original circuit by another circuit whose output does not depend on all input qubits.

Setup Consider a circuit on n qubits (no ancillæ) where the initial state is one of the 2^n standard basis states $\{|00\ldots0\rangle, |00\ldots1\rangle, \cdots, |11\ldots1\rangle\}$. Each of the possible initial states $\{|x\rangle \ : \ x \in \{0,1\}^n\}$ corresponds to the classical input x to the circuit. Henceforth any intermediate state of the circuit, denoted by $|\Psi(x)\rangle$, should be understood as the intermediate state when the input state is $|x\rangle$. Similarly, $\{|\Psi(x)\rangle\}$ will denote the distribution of intermediate states when the distribution of input states is $\{|x\rangle\}$.

Consider an intermediate state of the circuit, $|\Psi(x)\rangle$. We will analyse the effect of applying a $(t+1)$-qubit Toffoli gate $\mathcal{T}$ to this state. Without loss of generality, let us assume that $T = \{1, \ldots, t\}$ form the set of control qubits of $\mathcal{T}$ and its target qubit is qubit $t+1$. We will use $q = n - (t+1)$ to denote the number of qubits not involved with $\mathcal{T}$ and $f_x(j) = \langle j|\Psi(x)\rangle$ to denote the amplitude of $|j\rangle$ in $|\Psi(x)\rangle$. We will use the notation $i \cdot j$ to denote concatenation of strings.

First note that, for all x

$$\begin{aligned}
\Psi(x) &= \sum_{k\in\{0,1\}^n} f_x(k)|k\rangle \\
&= \sum_{j\in\{0,1\}^q} \sum_{i\in\{0,1\}^{t+1}} f_x(i\cdot j)|i\rangle|j\rangle \\
&= \sum_{j\in\{0,1\}^q} \Big[f_x(1^t\cdot 0\cdot j)|1^t\cdot 0\cdot j\rangle + f_x(1^t\cdot 1\cdot j)|1^t\cdot 1\cdot j\rangle + \sum_{\substack{i\neq 1^t\cdot 0\\ i\neq 1^t\cdot 1}} f_x(i\cdot j)|i\cdot j\rangle \Big]
\end{aligned}$$

After $\mathcal{T}$ is applied, the amplitudes of all but two of the basis states remain unchanged and the amplitudes of the remaining two basis states are swapped. We get the following set of states for all x.

$$\begin{aligned}
\mathcal{T}|\Psi(x)\rangle = &\sum_{j\in\{0,1\}^q} \Big[f_x(1^t\cdot 0\cdot j)\mathcal{T}|1^t\cdot 0\cdot j\rangle + f_x(1^t\cdot 1\cdot j)\mathcal{T}|1^t\cdot 1\cdot j\rangle + \\
&\qquad \sum_{\substack{i\neq 1^t\cdot 0\\ i\neq 1^t\cdot 1}} f_x(i\cdot j)\mathcal{T}|i\cdot j\rangle \Big] \\
= &\sum_{j\in\{0,1\}^q} \Big[f_x(1^t\cdot 0\cdot j)|1^t\cdot 1\cdot j\rangle + f_x(1^t\cdot 1\cdot j)|1^t\cdot 0\cdot j\rangle + \sum_{\substack{i\neq 1^t\cdot 0\\ i\neq 1^t\cdot 1}} f_x(i\cdot j)|i\cdot j\rangle \Big]
\end{aligned}$$

Here is a technical lemma regarding the amplitudes that will be useful later.

Lemma 4.4.2 *For any $y_1 \neq y_2 \in \{0,1\}^n$,*

$$\sum_x |f_x(y_1) - f_x(y_2)|^2 = 2$$

Proof. In this proof $\sum_x$ (and similarly $\sum_y$) will denote $\sum_{x \in \{0,1\}^n}$.

Let $C = \sum_x |\Psi(x)\rangle\langle x| = \sum_{x,y} f_x(y)|y\rangle\langle x|$ denote the operator that transforms $|x\rangle$ to $|\Psi(x)\rangle$. Since $|\Psi(x)\rangle$ is the intermediate state of the circuit when the initial state is $|x\rangle$, C is a unitary operator. Therefore, $C^\dagger = \sum_{xy} f_x(y)^*|x\rangle\langle y|$ is also unitary and hence, $\{C^\dagger|y\rangle \mid y \in \{0,1\}^n\}$ forms an orthonormal basis.

Therefore, for $y_1 \neq y_2$ $\|C^\dagger|y_1\rangle - C^\dagger|y_2\rangle\|^2 = 2$.

Since $\|C^\dagger|y_1\rangle - C^\dagger|y_2\rangle\|^2 = \|\sum_x (f_x(y_1)^* - f_x(y_2)^*)|x\rangle\|^2 = \sum_x |f_x(y_1) - f_x(y_2)|^2$, this completes the proof. □

Approximating a Toffoli gate Our strategy is to remove $\mathcal{T}$ (or replace it by the Identity gate), if t is large enough, and show that the expected error is reasonably low.

Lemma 4.4.3 *For a Toffoli gate $\mathcal{T}$ with t control qubits, acting on a state $|\Psi(x)\rangle$,*

$$\Pr_x [\|\mathcal{T}|\Psi(x)\rangle - |\Psi(x)\rangle\| < \epsilon] > 1 - \frac{2}{2^t\epsilon^2}$$

For proving this we need to define what we mean by approximation error. Consider a distribution over an ensemble of states $\Sigma = \{|\Psi_1\rangle, \ldots\}$.

Definition 4.4.4 *For a state $|\Psi\rangle$ and a gate $\mathcal{T}$, define the* expected error *in removing $\mathcal{T}$ as*

$$E_x(\Delta_{\mathcal{T}}^2) = \sum_i \Pr(|\Psi_i\rangle)\Delta_{\mathcal{T}}^2$$

where $\Pr(|\Psi_i\rangle)$ is the probability of choosing the state $|\Psi_i\rangle$ from Σ and $\Delta_{\mathcal{T}} = \|T|\Psi\rangle - |\Psi\rangle\|$.

In our specific case, we have a distribution on 2^n states $\{|\Psi(x)\rangle\}$. Since quantum gates are unitary and thereby reversible, this means that $\{|\Psi(x)\rangle\}$ is identically distributed as the input to the circuit $\{|x\rangle\}$, which we chose uniformly at random.

Next we need an estimation of the expected error $E_x(\Delta_{\mathcal{T}}^2)$.

Lemma 4.4.5 $E_x(\Delta_{\mathcal{T}}^2) = 2/2^t$

Proof. Notice that,

$$\mathcal{T}|\Psi(x)\rangle - |\Psi(x)\rangle = \sum_{j\in\{0,1\}^q} \Big[f_x(1^t\cdot 0\cdot j) - f_x(1^t\cdot 1\cdot j)\Big]\ |1^t\cdot 1\cdot j\rangle + \Big[f_x(1^t\cdot 1\cdot j) - f_x(1^t\cdot 0\cdot j)\Big]\ |1^t\cdot 0\cdot j\rangle$$

So, taking the difference and summing over all x gives us,

$$\begin{aligned}
\|\mathcal{T}|\Psi(x)\rangle - |\Psi(x)\rangle\|^2 &= \sum_{j\in\{0,1\}^q} 2|f_x(1^t\cdot 0\cdot j) - f_x(1^t\cdot 1\cdot j)|^2 \\
\sum_x \|\mathcal{T}|\Psi(x)\rangle - |\Psi(x)\rangle\|^2 &= \sum_x \sum_{j\in\{0,1\}^q} 2|f_x(1^t\cdot 0\cdot j) - f_x(1^t\cdot 1\cdot j)|^2 \\
&= 2\sum_{j\in\{0,1\}^q} \sum_x |f_x(1^t\cdot 0\cdot j) - f_x(1^t\cdot 1\cdot j)|^2 \\
&= 2\sum_{j\in\{0,1\}^q} 2 = 2^{q+2} = 2^{n-t+1} \qquad \text{(Using lemma 4.4.2)}
\end{aligned}$$

Therefore, $\sum_x \Delta_{\mathcal{T}}^2 = 2\cdot 2^n/2^t$ and since the distribution of $\{|\Psi(x)\rangle\}$ is uniform, this proves the lemma. □

Now we are ready to prove lemma 4.4.3.

Proof of Lemma 4.4.3 Using the Markov inequality we get,

$$\Pr_x[\Delta_{\mathcal{T}}^2 < \epsilon^2] \geq 1 - \frac{E_x(\Delta_{\mathcal{T}}^2)}{\epsilon^2}$$

The result follows after we replace the expected error from lemma 4.4.5,

$$\Pr_x[\Delta_{\mathcal{T}} < \epsilon] = \Pr_x[\Delta_{\mathcal{T}}^2 < \epsilon^2] > 1 - \frac{2}{2^t\epsilon^2}$$

□

4.4.3 Approximating a circuit

Lemma 4.4.3 gives us a bound on the probability of error when one Toffoli gate is removed from the circuit. We can do a similar estimation when several gates are removed.

The crucial fact to note here is that the states after each gate $|\Psi(x)\rangle$ have a one-to-one correspondence with the initial states $|x\rangle$ i.e. the $\{|x\rangle\}$ and $\{|\Psi(x)\rangle\}$ after each gate are identically distributed. The rest is a simple application of the union bound and using the fact that errors from quantum operations are additive.

Lemma 4.4.6 *Consider a circuit with $C = C_k\mathcal{T}_k\cdots C_1\mathcal{T}_1C_0$ with k Toffoli gates $\{\mathcal{T}_1,\ldots\}$ that are removed; C_i-s denote the intermediate gates that are not removed. Let t_i denote the number of control qubits $\mathcal{T}_i$ has. Then the total average error when all the k Toffoli gates are removed is upper bounded by the the sum of the average error for each gate, i.e.*

$$\Pr_x\left[\|C_k\mathcal{T}_kC_{k-1}\cdots C_1\mathcal{T}_1C_0|\Psi(x)\rangle - C_k\cdots C_0|\Psi(x)\rangle\| < \epsilon\right] > 1 - \frac{2\cdot k^2}{\epsilon^2}(1/2^{t_1}+\ldots+1/2^{t_k})$$

Proof. We will give a sketch of the proof for two consecutive gates. The general case where k Toffoli gates are removed and there are other intermediate gates is similar.

Say we have two Toffoli gates $\mathcal{T}_1$ and $\mathcal{T}_2$ with t_1 and t_2 control qubits respectively applied in succession to the state $|\Psi(x)\rangle$. From lemma 4.4.3 we know,

$$\Pr_x\left[\|\mathcal{T}_1|x\rangle - |x\rangle\| \geq \epsilon\right] < \frac{2}{2^{t_1}\epsilon^2}$$

$$\Pr_x\left[\|\mathcal{T}_2\mathcal{T}_1|x\rangle - \mathcal{T}_1|x\rangle\| \geq \epsilon\right] < \frac{2}{2^{t_2}\epsilon^2}$$

Due to additive nature of errors from quantum gates, $\|\mathcal{T}_2\mathcal{T}_1|x\rangle - |x\rangle\| \leq \|\mathcal{T}_2|\Psi(x)\rangle - |\Psi(x)\rangle\| + \|\mathcal{T}_1|\Psi(x)\rangle - |x\rangle\|$. By union bound,

$$\Pr_x\left[\|\mathcal{T}_1|\Psi(x)\rangle - |\Psi(x)\rangle\| \geq \epsilon \text{ or } \|\mathcal{T}_2\mathcal{T}_1|\Psi(x)\rangle - \mathcal{T}_1|\Psi(x)\rangle\| \geq \epsilon\right] < \frac{2}{\epsilon^2}(1/2^{t_1}+1/2^{t_2})$$

$$\Pr_x\left[\|\mathcal{T}_1|\Psi(x)\rangle - |\Psi(x)\rangle\| < \epsilon \text{ and } \|\mathcal{T}_2\mathcal{T}_1|\Psi(x)\rangle - \mathcal{T}_1|\Psi(x)\rangle\| < \epsilon\right] > 1 - \frac{2}{\epsilon^2}(1/2^{t_1}+1/2^{t_2})$$

$$\Pr_x\left[\|\mathcal{T}_2\mathcal{T}_1|\Psi(x)\rangle - |\Psi(x)\rangle\| < 2\epsilon\right] > 1 - \frac{2}{\epsilon^2}(1/2^{t_1}+1/2^{t_2})$$

Replacing 2ϵ by ϵ proves the two gate case. □

To approximate a circuit of depth d and on n inputs without any ancillæ, we start from the input level of the circuit, remove *large* gates, move to the next higher level and then repeat the same. At each level we remove Toffoli gates with t or more control qubits;

$t = t(n, d)$ can be a fixed parameter per circuit family to control the error. This gives us our main theorem characterising such circuits.

Theorem 4.4.7 *Let C be a quantum circuit with n qubits, no ancillæ depth d, and using unbounded fanin Toffoli gates and bounded fanin gates. For any $t < n$, let S_t be the number of gates with fanin at least $t + 1$. Then, there exists another quantum circuit C' on n qubits, no ancillæ, with depth at most d and using the same set of gates such that C' only uses gates with fanin at most t and*

$$\Pr_{x \in \{0,1\}^n} \left[\| C|x\rangle - C'|x\rangle \| < \epsilon \right] > 1 - \frac{2S_t^3}{\epsilon^2 2^t}$$

Approximating a layer We can get a slightly better error bound by observing that in any given layer, the gates, especially the Toffoli gates that are removed, operate on disjoint sets of qubits. That is, if all the k gates removed in lemma 4.4.6 are of the same layer then the error is less than just the sum of the errors. We are presenting this after the main theorem since it is easier to understand how to approximate a layer once the general idea is clear.

Consider a layer of gates in a circuit C: let $\mathcal{T}_1, \ldots, \mathcal{T}_k$ denote k Toffoli gates in L that will be removed and let t_i denote the number of control qubits of $\mathcal{T}_i$. Let $\mathcal{T}_0$ denote the combined unitary gate for the rest of the qubits. To keep the notation simple, we will let $\mathcal{T}_0$ operate also on untouched qubits in this layer so that T_0 can be applied to $t_0 = n - \sum_i t_i$ qubits. Let L denote the unitary operator for this layer: $L = \mathcal{T}_k \otimes \ldots \mathcal{T}_1 \otimes \mathcal{T}_0$.

As before, denote the error in approximating the layer as $\Delta_L = \| L|\Psi(x)\rangle - |\Psi(x)\rangle \|$. The expected error for this layer L is defined similarly as before: $E_x(\Delta_L^2) = \sum_x \left(\frac{1}{2^n}\right) \Delta_L^2$.

Lemma 4.4.8 *The expected error for a layer with k Toffoli gates, where the ith gate has t_i control qubits, is*

$$E_x(\Delta_L^2) = 2 \left(1 - \prod_{i=1}^{k} \left(1 - \frac{1}{2^{t_i}} \right) \right)$$

Proof. We will follow the lines of lemma 4.4.5. To keep the notation simple, let T_1 act on qubits $1, \ldots, (t_1 + 1)$, T_2 act on qubits $(t_1 + 2), \ldots$ and so on; T_0 acts on the last q qubits.

Then, we can write

$$|\Psi(x)\rangle = \sum_{i_0,i_1,\ldots} f_x(i_1 \cdot i_2 \ldots i_0)|i_1 \otimes i_2 \otimes \ldots i_k \otimes i_0\rangle$$

where $i_0 \in \{0,1\}^q$ and $i_j \in \{0,1\}^{t_j}$ for $j = 1 \ldots k$ represent the basis states of the set of qubits each Toffoli gate acts on.

When any of the Toffoli gates of L is applied on one of the basis states, the amplitude of the state changes if and only if the all its control qubits are in the state $|1\rangle$. In fact, as shown earlier, the amplitudes of such states with target qubit in the states $|0\rangle$ and $|1\rangle$ are swapped i.e. the amplitudes undergo a permutation. So we can write the state after L is applied as

$$L|\Psi(x)\rangle = \sum_{i_0,i_1,\ldots} f_x(i'_1 \cdot i'_2 \ldots i'_0)|i_1 \otimes i_2 \otimes \ldots i_k \otimes i_0\rangle$$

where, for $j = 1 \ldots k$, $i'_j \neq i_j$ *if and only if* i_j is one of the two values $11\ldots10$ and $11\ldots11$ (in which case i'_j is the same as i_j with the last bit flipped).

The number of possible states $|i_1 \cdot i_2 \ldots i_0\rangle$ whose amplitude is thus permuted is

$$2^{t_0}\prod_{j=1}^{k} 2^{t_j+1} - 2^{t_0}\prod_{j=1}^{k}(2^{t_j+1} - 2) \tag{4.1}$$

Doing a similar analysis as in lemma 4.4.5,

$$\sum_x \|L|\Psi(x)\rangle - |\Psi(x)\rangle\| =$$
$$2\sum_{i_0,i_j,\ldots}\sum_x |f_x\left(\underbrace{\tilde{i_1}\cdot 0}_{i_1}\cdot\underbrace{\tilde{i_2}\cdot 0}_{i_2}\ldots\underbrace{\tilde{i_k}\cdot 0}_{i_k}\ldots\cdot i_0\right) - f_x\left(\underbrace{\tilde{i_1}\cdot 1}_{i_1}\cdot\underbrace{\tilde{i_2}\cdot 1}_{i_2}\ldots\underbrace{\tilde{i_k}\cdot 1}_{i_k}\cdot i_0\right)|^2$$

for each choice of $i_1, i_2, \ldots$ such that at least some i_j has $\tilde{i_j} = 11\ldots1$. From lemma 4.4.2, each term in the above sum is 2 and the number of possible values of $i_0, i_1, \ldots$ is given by equation 4.1. Since the distribution of the input $|x\rangle$ (and hence $|\Psi(x)\rangle$) is uniform, we get

the required expression for the expected error.

$$\begin{aligned} E_x(\Delta_L^2) &= \frac{1}{2^n} 2 \left[2^{t_0} \prod_{j=1}^{k} 2^{t_j+1} - 2^{t_0} \prod_{j=1}^{k} (2^{t_j+1} - 2) \right] \\ &= \frac{2}{\prod_{j=1}^{k} 2^{t_j+1}} \left[\prod_{j=1}^{k} 2^{t_j+1} - \prod_{j=1}^{k} (2^{t_j+1} - 2) \right] \\ &= \frac{2}{\prod_{j=1}^{k} 2^{t_j}} \left[\prod_{j=1}^{k} 2^{t_j} - \prod_{j=1}^{k} (2^{t_j} - 1) \right] \\ &= 2 \left(1 - \prod_{i=1}^{k} \left(1 - \frac{1}{2^{t_i}} \right) \right) \end{aligned}$$

□

We can now approximate a circuit layer by layer, giving us a modified version of Theorem 4.4.7. Assume we remove all Toffoli gates with t or more control qubits. Let $E(i) = 2 - 2\left(1 - \frac{1}{2^t}\right)^i$ denote an upper bound for the expected error for any layer where i Toffoli gates are removed.

Corollary 4.4.9 *Let C be a quantum circuit with n qubits, no ancillæ and has depth d and uses unbounded fanin Toffoli gates and bounded fanin gates. For any $t < n$, let d_i be the number of Toffoli gates with fanin at least $t+1$ in layer i (count the output layer as layer 1). Then, there exists another quantum circuit C' on n qubits, no ancillæ, with depth at most d and using the same set of gates such that C' only uses gates with fanin at most t and*

$$\Pr_x \left[\| C|x\rangle - C'|x\rangle \| < \epsilon \right] > 1 - \frac{d^2}{\epsilon^2} \sum_{i=1}^{d} E(d_i)$$

4.5 Application: Lower bound for parity

We will use the technique above to give another proof that constant depth quantum circuits without ancillæ cannot compute parity using only unbounded fanin Toffoli gates and single qubit gates. As mentioned earlier, this was first proved in [FFG+06], where they were also able to allow sublinear number of ancillæ.

We first need a technical lemma which shows that if two families of quantum states are *close* in terms of Euclidean distance, then the distribution of their measurement outcomes are also *close*. A general proof of this requires the concept of distance measures for quantum states (e.g. trace distance), which are beyond the scope of this thesis [1]; here we state and prove a simpler version of the lemma that is immediately useful to us.

Lemma 4.5.1 *Consider two pure states over n qubits,* $|\Psi\rangle = |0\rangle \otimes |\Psi_0\rangle$ *and* $|\Phi\rangle = |0\rangle \otimes |\Phi_0\rangle + |1\rangle \otimes |\Phi_1\rangle$ *(*$|\Phi_0\rangle$ *and* $|\Phi_1\rangle$ *may be unnormalised) where the latter is a close approximation of the former in the sense that* $\||\Psi\rangle - |\Phi\rangle\| < \epsilon$ *for some small* ϵ*.*

Let p_0 *and* q_0 *denote the probability of measuring* 0 *in the first qubit of* $|\Psi\rangle$ *and* $|\Phi\rangle$ *respectively (use any set of measurement operators that distinguish between a 0 and 1 in the first qubit). Then,* $|p_0 - q_0| \leq \epsilon^2$*.*

Proof. p_0 is obviously 1 and since $q_0 = \||\Phi_0\rangle\|^2$, so we will only have to prove that $\||\Phi_0\rangle\|^2 > 1 - \epsilon^2$.

$$\begin{aligned}
\epsilon^2 &> \||\Psi\rangle - |\Phi\rangle\|^2 \\
&= \||0\rangle \otimes (|\Psi_0\rangle - |\Phi_0\rangle) + |1\rangle \otimes |\Phi_1\rangle\|^2 \\
&= \||\Psi_0\rangle - |\Phi_0\rangle\|^2 + \||\Phi_1\rangle\|^2 \\
&= \||\Psi_0\rangle\|^2 + \||\Phi_0\rangle\|^2 + \||\Phi_1\rangle\|^2 - \langle\Psi_0|\Phi_0\rangle - \langle\Phi_0|\Psi_0\rangle \\
&= 2 - 2\mathrm{Re}(\langle\Psi_0|\Phi_0\rangle) \\
&\geq 2 - 2|\langle\Psi_0|\Phi_0\rangle| \\
&\geq 2 - 2\||\Psi_0\rangle\| \cdot \||\Phi_0\rangle\| \quad \text{(by Cauchy-Schwarz inequality)} \\
&= 2 - 2\||\Phi_0\rangle\| \\
\||\Phi_0\rangle\|^2 &> (1 - \epsilon^2/2)^2 > 1 - \epsilon^2
\end{aligned}$$

[1] The general statement is if $|\Psi\rangle$ and $|\Phi\rangle$ have an Euclidean distance at most ϵ, then for any set of measurement operators, the respective probabilities of obtaining a particular outcome differ by at most 2ϵ. For this and other details on distance measures in quantum computing, see [NC00, KSV02].

□

The following theorem shows that no constant depth quantum circuit (of the kind we consider here) can compute parity. In fact, we prove that parity cannot be computed even when the depth is arbitrarily close to, but strictly less than, $\log n$. This is close to the upper bound since parity can be trivially computed (uncleanly) in depth at most $\log n$.

Theorem 4.5.2 *No quantum circuit of depth d can compute parity exactly using only single qubit gates and Toffoli gates any without any ancillæ when $d = O(\log n)^{1-c}$ for any $c > 0$.*

Proof. Assume on the contrary, that there is a circuit C which can compute parity and is of the type mentioned in the theorem; let d denote its depth and n its number of inputs. We will use Theorem 4.4.7 on C to obtain an approximate circuit C' and then argue that such a C cannot exist. We will use $\oplus(x)$ to denote the parity of n-bit x and δ to denote the term $\frac{2S_t^3}{\epsilon^2 2^t}$ appearing in the theorem.

Note that C is a circuit with an exact output i.e. measuring the output qubit of $C|x\rangle$ gives $\oplus(x)$ for all x. Now, using the above mentioned theorem, we can obtain an approximate circuit C' such that the for at least $(1-\delta)$ fractions of all 2^n possible inputs x, $\|C|x\rangle - C'|x\rangle\| < \epsilon$. For all such x, lemma 4.5.1 tells us that the measurement outcome of $C'|x\rangle$ will be $\oplus(x)$ with probability at least $(1-\epsilon^2)$. After C' is applied on any input, and the output state measured in the standard basis, the output qubit will contain the correct value with probability at least $(1-\epsilon^2)(1-\delta)$.

All gates in C' will have fanin at most t; so if $t^{d-1} < n$, then no qubit at the final layer, especially the measurement qubit, can be connected to all n input qubits. We will choose a suitable $t < n^{1/(d-1)}$ and an ϵ such that $(1-\epsilon^2)(1-\delta) > 1/2$. It is well known that, information theoretically it is not possible to predict the parity of n bits with probability more than $1/2$ even if only 1 bit is ignored. Since the output qubit in the modified circuit will not be connected to at least one qubit, this will imply that no such C can exist and the theorem will be proved.

To finish the proof we only have to find suitable values of t and ϵ and estimate S_t. There are at most n/t large Toffoli gates in each layer. Thus at most $S_t = nd/t$ Toffoli gates are removed in C'. Though using this value of S gives us our desired result, we will do a more careful counting in the interest of rigour.

Define a gate to be *large* if it has fanin more than t. We will modify C to create C' by removing gates starting from the output layer to the input layer. Start from the output layer; if the output qubit is connected to a large Toffoli gate, remove this gate. Now consider the next layer in this modified circuit; at most t qubits in the layer could be connected to the output qubit. Consider the gates acting on these qubits and if any of them are large, remove them. Continuing like this, we will remove at most $1 + t + t^2 + \ldots + t^{d-1} = \frac{t^d-1}{t-1} < \frac{2n}{t}$ large Toffoli gates and obtain circuit C'. Choose $S_t = 2n/t$. However, the modified circuit might still have large Toffoli gates; but since none of them is on the path to the output qubit, in the next step also remove those gates and obtain another circuit C'' all of whose gates are of fanin at most t. Using the above arguments, measuring the output qubit will give the same value as that of C with probability at least $(1-\epsilon)(1-\delta)$. On the other hand, using a communication argument as in Theorem 4.3.1, the measurement output of C' is exactly the same as that of C''. So, we obtain a modified circuit C'' which does not use any large gate and whose output matches that of C with probability at least $(1-\epsilon)(1-\frac{2\cdot 8n^3}{\epsilon^2 2^t t^3})$.

For constant depth circuits, d would be a constant. Choosing $t = 3\lg n < n^{1/(d-1)}$ and $\epsilon = 1/4$, we have,

$$(1-\epsilon^2)(1-\delta) = \frac{15}{16}(1-\frac{29}{\lg^3 n}) > 1/2$$

for large enough n.

To allow for larger depth, consider any $d < (\log n)^{1-c}$ for any constant $c > 0$. Let $n = 2^m$. Then, $d < m^{1-c} = m/m^c$ and $n^{1/d} > 2^{m^c}$. Choose $t = n^{1/2d} > 2^{\frac{m^c}{2}}$ (note that $t < n^{1/(d-1)}$). Since $\log n = m \in o\left((\sqrt{2})^{m^c}\right)$, we get $n^3 \in o(2^t)$. Therefore, $\frac{n^3}{t^3 2^t} \in o(1)$. We can choose any small ϵ e.g. $1/4$ and that will give us $(1-\epsilon)(1-\delta) > 3/4(1-o(1)) > 1/2$.

□

Different lower bounds for parity exploit a different property of this function. Accordingly, a particular technique works also for other functions with similar properties. Apart from the parity function, our technique may be useful against any function f with the following property:

f cannot be approximated for a reasonably large fraction of its inputs by any function g that does not depend on at least one of its input bits.

The only other known quantum circuit lower bound technique, by Fang et al, relates the minimum depth required to the minimum sensitivity of a Boolean function, denoted by $s_{\min}(f)$. $s_{\min}(f) = m$ if for all inputs to f, there are at least m bit positions flipping whom will flip the output of f [2]. They show that any clean quantum circuit whose output is exactly $f(x)$ must have depth at least $1.44 \log(s_{\min}(f)) - 1$.

4.6 Summary

In this chapter we looked at quantum circuits without ancillæ made of any set of bounded fanin gates and unbounded fanin Toffoli gates. It has been shown earlier that this set of gates is powerful enough to compute a large class of functions. If we leave out the unbounded fanin gates, the circuits we can construct are provably not as versatile; they cannot compute any non-trivial function that depends on all input bits. The Toffoli gate being essentially a classical gate, we wanted to investigate how much power does it add to constant fanin quantum circuits. We showed that we can remove Toffoli gates of up to a certain fanin while still giving approximately the correct output.

The obvious open question is how to use this technique for circuits with ancillæ. Fang et al. [FFG+06] showed how to get a similar lower bound for quantum circuits with a sublinear number of ancillæ. However we believe that the lower bound is true for

[2] For $x \in \{0,1\}^n$, let x^i denote x with its ith bit flipped. Then formally, $s_{\min}(f) = \min_x |\{i \mid f(x) \neq f(x^i)\}|$. This is different than the usual notion of sensitivity of f [Nis89]: $s(f) = \max_x |\{i \mid f(x) \neq f(x^i)\}|$. We will revisit this notion in Chapter 5.

unbounded number of ancillæ. While it is tempting to use the technique discussed here for circuits with ancillæ, it is not immediately clear what constitutes a *large gate*. This is because, with ancillæ, the states of the qubits can be copied and lemma 4.4.5 does not hold any more.

For classical circuits, the question whether constant depth unbounded fanin AND, OR, NOT circuits can compute parity was open for a long time before it was proved to be impossible using several different but sophisticated methods. However, if we restrict the classical circuits to not use any fanout, simple versions of these techniques can prove the lower bound. Both our technique and the earlier technique for dealing with constant depth quantum circuits without ancillæ (without any ancillæ, there cannot be any fanout) are based on ideas similar to the classical circuit techniques but are quite complicated otherwise. Extending them to the general case seems difficult but we are still hopeful. We would like to end this chapter with a conjecture that is the quantum analogue of the classical parity problem.

Conjecture 4.6.1 *No quantum circuit on n inputs and of depth $o(\log n)$ can compute parity using bounded fanin gates and unbounded fanin Toffoli gates.*

5 Detecting Input Sensitivity of Gates

"Good afternoon, gentlemen. I am a Hal 9000 computer. . . . I am completely operational, and all my circuits are functioning perfectly."

– Broken H.A.L. 9000 *in* "2001: A Space Odyssey"

Faults appear everywhere, disregarding our approval. It is no surprise that the wires, gates, chips that make up a hardware circuit are subject to high degree of a faults, given their small size and complex manufacturing process. One usual way of diagnosing faulty components is to summon them to a lie-detector test. Surprisingly (and thankfully), hardware components are not smart enough to cheat, so the odd player can be easily detected and benched; well ... most of them.

In the last two chapters we have almost pitched two complex quantum gates against each other; the Toffoli gate (computing the AND function) and the fanout gate (equivalently, the parity gate computing the Parity function), with some evidence that the latter is more complicated than the former. It is usually easier to find any outlier if the rules of a game are strict. Does that mean in our case, and very very informally, that it is easier to detect faults in gates that compute *harder* functions? The original motivation behind this chapter, based on the following report, was to compare the hardness of verifying particular kind of faulty classical and quantum Boolean gates computing the AND and the Parity function and whether quantum parallelism can buy us any improvement.

•[BH09] D. Bera, S. Homer. On Finding Sensitivity of Quantum and Classical Gates.

Boston University, Computer Science technical report 2009-019 (2009).

5.1 Introduction

Like all hardware, quantum gates and circuits are prone to faults. In fact, implementations of quantum gates are extremely sensitive to various physical factors and often have errors and faults of various kinds. There has been extensive research on fault-tolerant quantum computing and detecting/correcting errors in a quantum computation [Sho96]. Our motivation is in a similar spirit but in a tangential direction. We are interested in exploiting *quantum parallelism* to identify certain types of errors in specific gates and circuits more efficiently than is possible using classical techniques. Our focus is not on arbitrary errors or arbitrary gates, but rather on a certain type of error for several specific, simple quantum operations. However, the problem we tackle has a close resemblance to the fault models used for the fault testing of classical Boolean circuits. We elaborate on this in section 5.3.

We will consider n-bit Boolean gates with one output. Usually G depends on all n of its inputs. However for this chapter consider gates which may be independent of some of these inputs, for example, if the connection from some input to the gates output is broken. We would like to determine which inputs it depends on or at least if there is some *faulty input* which it does not depend on at all. Here G could be a simple Boolean gate like AND or OR, or some complicated Boolean circuit with one output; we are merely interested in the input-output behaviour of the gate.

Assume we are given a few samples of a gate which we want to test[1]. How easy or hard it is to find out whether G is faulty or not (and if possible, which inputs are not connected)? We want to investigate this question by considering classical and quantum algorithms which are allowed to query copies of the faulty gates. We measure the number

[1]Yes, it is usually unreasonable to assume multiple exact copies of the same faulty gate; faults are not so well-behaved to be consistent across gates. But in our theoretical study on the properties of the fault and the gates, we will assume that our faulty gates are identically faulty. Moreover, section 5.3 tries to explain some cases where this might be possible.

of *queries to the faulty gates* needed to determine the faulty input(s). Notice that this problem is similar to algebraic query complexity, where, instead of using a bit of the input, the nodes of a decision tree are labelled by the output of an algebraic function of the input. Using similar ideas we extend existing quantum query results to obtain upper and lower bounds for detecting faults in AND and other similar gates.

We claim that quantum algorithms can diagnose faulty gates and determine their faulty inputs more efficiently than classical ones. We will illustrate this for several common and central types of gates, the Parity gate and the AND gate among them. For the quantum case, we will construct quantum circuits which will be given the usual quantum analogue of these gates. The output of the checking circuits will indicate one or all the faults of the gate in question. Since a classical circuit usually has a single output whereas a quantum circuit has multiple outputs, for the classical case we show the lower bound for a simpler version of the problem: Detecting whether there is *any* faulty input.

5.2 Problem statement

We first define what we mean by a faulty classical gate in this chapter. The faulty quantum gates used in this chapter will be their reversible extensions [2].

Definition 5.2.1 (Faulty gate) *Consider a gate G computing a Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$. We say that the* gate has at least s faults *if there exists a set of "faulty" inputs $\mathcal{S} \subseteq \{1,\ldots,n\}$ of size s such that $\forall (x_1,\ldots,x_n) \in \{0,1\}^n,\ \forall (x'_1,\ldots,x'_n) \in \{0,1\}^n,\ f(x_1,\ldots,x_n) = f(x'_1,\ldots,x'_n)$ whenever $\forall i \notin \mathcal{S},\ x_i = x'_i$ (the value of f depends only on the non-faulty inputs irrespective of the inputs from $\mathcal{S}$).*

We need to be careful in dealing with such faulty gates since the functions computed by the faulty gate and the correct gate might be of a completely different nature. Certain

[2] A reversible way to compute a function f by a quantum gate is $|x_1,\ldots,x_n,y\rangle \rightarrow |x_1,\ldots,x_n,y \oplus f(x_1,\ldots,x_n)\rangle$

functions do not have a natural restriction when defined only on a subset of its inputs; e.g., the function $f(x_1, x_2, x_3) = x_1 \oplus x_2 x_3$ has no obvious unique restriction to only the first and the second inputs. On the other hand, for the function $f(x_1, x_2, x_3) = x_1 \vee x_2 \vee x_3$, a natural way to define its faulty version could be $f(x_1, x_2, x_3) = x_1 \vee x_2$ (assuming the third input is faulty). In reality the faulty function in both the cases could be defined in a non-trivial way (e.g. depending on the actual hardware implementation). A Boolean function is called as a *symmetric* function if the output depends only on the number of ones in the input (not the specific positions of the ones). One way to define faulty versions of a gate computing a symmetric function could be to assume that if some input wires are not connected, then the gate computes the natural restriction of the function on the rest of its inputs.

For a given gate, consider the largest possible set of faulty inputs; in the rest of this chapter, we will denote the maximal set by $\mathcal{S}$ and its size by s. Note that, by our definition, any subset of $\mathcal{S}$ is a valid set of faulty inputs to the gate. We will use e to denote an elementary vector. Let e_i denote the i-th elementary n-bit string (only the i-th bit is set to 1, other $n-1$ bits are set to 0) and $f_i = \overline{e_i}$ (bit-wise complement). To allow our quantum circuits to directly output the faulty inputs in an easy manner, we will interchangeably use another representation of S using elementary vectors: $\{e_i \mid i \in \mathcal{S}\}$ (when represented using elementary vectors, $e_i \in \mathcal{S}$ should be interpreted to mean that the i-th input is faulty). We will use $\sigma \in \{0,1\}^n$ to denote the characteristic vector of $\mathcal{S}$: $\sigma_i = 1$ if and only if $e_i \in \mathcal{S}$. We will use the following notation in this chapter: $\exists^l x$ indicates l distinct values of x.

Our goal is to construct oracle circuits using these faulty gates to detect faults. The cost of the detection is measured by the number of faulty gates queried. The detecting circuits can output different kind of information:

1. A single bit Boolean output denoting if there is any fault or not (whether $\mathcal{S} \stackrel{?}{=} \emptyset$).

2. An n-bit output representing any of the faulty inputs (output any elementary vector

$e \in \mathcal{S}$).

3. The output which indicates the maximal set of faulty inputs (that is, the n-bit binary string σ).

Note that, 1 can be reduced to 2 and 2 can be reduced to 3. Since classical gates have only one output, a classical circuit with n-bit output will inevitably require n gates. Thus, we show lower bounds for classical circuits of type 1. The quantum circuits we construct are of type 2 or 3.

The output can be deterministic or probabilistic; we cover both cases here. The two functions we will extensively deal with in this chapter are the parity function $\oplus(x_1, \ldots, x_n) = x_1 \oplus \ldots \oplus x_n$, for which we consider deterministic detecting circuits and the AND function $AND(x_1, \ldots, x_n) = x_1 \wedge \ldots \wedge x_n$, for which we consider probabilistic detecting circuits.

5.2.1 Relation to sensitivity

Our definition of faulty inputs has a close similarity to the sensitivity of Boolean functions [Nis89, BW02]. Let x denote an n-bit Boolean string and x^i is obtained from x by flipping x_i.

Definition 5.2.2 (Sensitivity) *An n-bit Boolean function f is sensitive to the i-th input bit on input x if $f(x) \neq f(x^i)$. The sensitivity of f on x, $s_x(f)$, is the number of bits to which f is sensitive on x.*
f is sensitive to i if there is some x such that f is sensitive to i on x. The sensitivity *of f is $s(f) = \max_x s_x(f)$.*
The insensitivity *of f, denoted by $\bar{s}(f)$, is the maximum number of i such that f is not sensitive to i.*

So, f is *insensitive* to the i-th input if f is insensitive to i on x for all x. If there is a gate G_f computing f where then f is insensitive to the i-th input, we say that the i-th

input to G is faulty. From this perspective, in this chapter we are trying to determine the input insensitivity of f using one or more copies of G_f. Note that for any n-bit function f, $s(f) = n - \bar{s}(f)$, so determining the insensitivity also determines the sensitivity of f.

Suppose we are allowed to query an n bit Boolean gate G_f to determine the input sensitivity of the Boolean function f. Obviously we need $\Omega(n)$ queries. The hardest case is when we have no *à priori* knowledge about f; it can be one of the 2^{2^n} possible functions. We can determine if f depends on any particular bit by querying f on all 2^n inputs; we believe that this is a tight upper bound.

However, we can do better if we know the possible ways f might behave when it is not reading all its inputs. Suppose f is insensitive to the input bits in the subset $s \subseteq \{1, \ldots, n\}$. For certain functions, we might have information how f might behave given such an s e.g. if f is supposed to compute the AND function, then we might be told that f actually computes the AND of the inputs in $\bar{s}$. In such cases, f belongs to a set of 2^n possible functions, one for each possible s. We consider two such functions here, the AND and the Parity function and show that it is possible to determine the sensitivity using $O(n)$ queries. Furthermore, we show that for some functions we can use quantum techniques to determine the sensitivity more efficiently than is possible using classical algorithms.

5.2.2 Organisation

The rest of this chapter is organised in the following manner. In the following section, we present some related work. We discuss our results on Parity gates in Section 5.4 and our results on AND gates in Section 5.6. For both the gates we present a quantum algorithm to detect faults and prove that it is optimal. We also prove that the quantum algorithm is provably better than any classical algorithm for detecting faults.

The construction and lower bounds for the AND gate will use techniques from query complexity. The proofs for the quantum case require extending a few quantum query complexity results to work with generalised oracles, which we discuss first in section 5.5 and then describe the actual quantum upper bound in Subsection 5.6.1 and matching

lower bounds in Subsection 5.6.2.

5.3 Related work

Boolean circuit testing is a well-established discipline within semiconductor circuit design. Over time circuits have increased in complexity and decreased in size, and the testing regimes have gotten more difficult and expensive. As they become *really* small quantum effects start to appear and need to be addressed. Recently testing methods have been suggested which use quantum circuits designed to find and analyse faults in circuits which compute classical Boolean functions. Exploiting the advantages of quantum techniques, these quantum circuits can be tested more easily and efficiently than their classical counterparts [CTK08].

Real world hardware circuits can fail in a multitude of ways. To deal with the variety of faults, the circuit testing research community has created different logical fault models. One of the most widely used logical fault models to represent faulty interconnections is the *stuck-at* model. In this model, one or more wires is assumed to be stuck at a fixed logic value, either 0 or 1 (respectively known as stuck-at-0 and stuck-at-1 model) [3]. The function computed by the corresponding gate then does not depend on the input carried by the faulty wire. This model captures several physical defects, e.g. a short-circuit between the wire and the ground, a disconnected wire or some internal fault which always sets the wire to a constant value.

Our problem is similar to the stuck-at fault model; we are dealing with circuits in which some wire may not be connected to its gate. This is similar to the situation where the wire maybe assumed to be stuck at some fixed value. However, our approach differs from the standard technique of detecting such faults (see e.g. [PBL05]); instead of generating a test set of input vectors whose output is then matched with the expected outputs, we design a circuit/algorithm using one or more faulty gates and expect to obtain infor-

[3] http://www.pld.ttu.ee/diagnostika/theory/fault.html

mation about the faulty inputs.

There is yet another difference with the classical case. An open wire in a classical reversible circuit can be modelled using multiple stuck-at faults. It has been shown that multiple stuck-at faults are functionally identical to single stuck-at faults for classical reversible circuits [PHM04]. Though the technique described in [CTK08] can detect stuck-at faults in quantum Boolean circuits, it cannot detect open wires. This is one piece of evidence that classical fault models might not directly work for quantum circuits.

Arvind *et al.* looked at a similar problem in [ASV98] where they studied program checking (as defined by Blum) using AC^0 circuits as checkers. Like us, they allowed their checkers to make queries to the program and defined the cost of checking as the number of queries. Note that they considered programs and not fixed input gates and they studied deterministic and randomised checkers for P-complete and NC^1-complete problems.

5.4 Parity gate

For this section, we will consider *faulty parity gates* defined by the function $\oplus_{\mathcal{S}}(x_1, \ldots, x_n) = \bigoplus_{i \notin \mathcal{S}} x_i$, where $\mathcal{S}$ is the set of faulty inputs. It will be useful to adopt a convention that $\oplus_{\{1,2\ldots,n\}}(x_1, \ldots, x_n) = 0$. The corresponding quantum gate U is the canonical way to reversibly compute the parity function:

$$U|x_1, \ldots, x_n, y\rangle = |x_1, \ldots, x_n, y \oplus \left(\textstyle\bigoplus_{\mathcal{S}}(x_1, \ldots, x_n)\right)\rangle$$

There is an obvious $O(n)$ algorithm to find all the faults of a parity gate: Query the gate on $e_1, \ldots, e_n$. The ith input is faulty if the output for e_i is 0. In contrast, we will present a quantum circuit which will output the all the faults of such a quantum parity gate using only one query. We will also show that this is significantly better than what classical circuits can do; a classical circuit requires $\Omega(n)$ queries even to detect if there is a faulty input or not. We will also show that even a bounded error algorithm cannot do any better than the trivial algorithm presented above.

5.4.1 Quantum detection circuit

Quantum parity gates could be checked by a depth 3, linear size quantum circuit using only one parity gate.

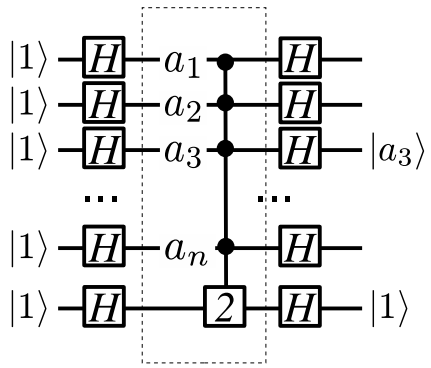

Figure 5.1: Circuit to check a faulty parity gate. a_i' is 1 if $i \in \mathcal{S}$, 0 otherwise.

The circuit to check a parity gate is given in Figure 5.1. A parity gate with Hadamard gates on all inputs on both the sides was shown equivalent to a quantum fanout gate where the target qubit of the parity gate becomes the control qubit of the fanout gate [Moo99]. Hence, if input a_i is not faulty, $|a_i'\rangle = |0\rangle$. The qubits in the faulty input will be unaffected; for these qubits, $|a_i'\rangle = |1\rangle$. Thus, the output of the circuit will be exactly σ, the characteristic vector of the fault set.

Theorem 5.4.1 *For a (possibly) faulty quantum parity gate $\bigoplus$ on n inputs, there is a quantum circuit on n inputs, with linear size, having depth 3, and making one call to the faulty gate, whose output qubit i measures* 1 *if the i-th input is faulty and* 0 *otherwise.*

5.4.2 Classical deterministic lower bound

Here we show that even detecting if a given parity gate is faulty or not, requires n queries classically. Consider the decision tree of a detecting circuit, C, which makes k queries. We will construct two parity gates G_S and G_T with fault sets $S \neq \emptyset$ and $T = \emptyset$ respectively and show that for C to distinguish between G_S and G_T, $k \geq n$.

We will construct S inductively based on the decision tree queries. Let S_j be the fault

set we construct after the j-th query by the decision tree (so $S = S_k$). We start with $S_0 = \{1, \ldots, n\}$ (all inputs are faulty). Now consider the $j+1$-th query. Say C queries $G_*(x_1, \ldots, x_n)$ ($*$ denotes either T or S). Let $b = \oplus_i x_i$; note that, $G_T(x_1, \ldots, x_n) = b$ and $G_S(x_1, \ldots, x_n) = \bigoplus_{i \notin S} x_i$.

Case 1: $G_{S_j}(x_1, \ldots, x_n) = b$. Set $S_{j+1} = S_j$. So, both G_T and $G_{S_{j+1}}$ are consistent after $1, \ldots, j+1$-th queries.

Case 2: $G_{S_j}(x_1, \ldots, x_n) = \overline{b}$. Then, $\exists c \in S_j,\ x_c = 1$. Set $S_{j+1} = S_j \setminus \{c\}$. Again, both G_T and $G_{S_{j+1}}$ are consistent after $1, \ldots, j+1$-th queries.

So, after k queries, $|S_k| \geq n - k$. If $k < n$, then C cannot distinguish between G_S and G_T.

Theorem 5.4.2 *Consider any classical Boolean circuit C which can also make queries to a (possibly) faulty $\bigoplus$ gate on n-inputs. Then there are two such gates $\bigoplus_1$ and $\bigoplus_2$ which C cannot distinguish without making n queries.*

5.4.3 Classical probabilistic lower bound

We showed how to construct a quantum checker that deterministically finds all faults of a parity gate using only 1 query. We claimed that this is order of magnitude better than what a classical checker can do and as evidence, argued that any deterministic classical checker using less than n queries is incapable of finding all the faults of a parity gate. However it is conceivable that a probabilistic classical checker might be able to perform better than a classical checker. In this section, we negate this possibility; we show that any probabilistic checker requires $\Omega(n)$ queries even with two-sided error.

For the lower bound, we relate the fault detection of a parity gate to a property testing problem discussed by Buhrman *et al.* in [BFNR08].

Let $y \cdot i$ denote the inner product of two vectors $y, i \in \mathcal{F}_2^n$ and for any $A \subseteq \{0,1\}^n$, let $P_A = \{x \ :\ \exists y \in A,\ \forall i \in \{0,1\}^n, x_i = y \cdot i\}$ denote the set of strings which represent all possible oracle replies for any string in A. Then P_A has an ϵ-tester with q queries if there

exists a query algorithm M which on input a 2^n-bit x, makes at most q queries to obtain individual bits of x, such that

1. The tester accept any input with the property: If $x \in P_A$, then $\Pr(M \text{ accepts}) \geq 2/3$
2. The tester rejects all inputs that are far from the property:
   $\forall z \in P_A$, if $|\{i \ : \ x_i \neq z_i\}| \geq \epsilon 2^n$, then $\Pr(M \text{ rejects}) \leq 1/3$

Theorem 5.4.3 (Lemma 3,[BFNR08]) *For most $A \subseteq \{0,1\}^n$, any ϵ-tester (for $\epsilon \leq 1/2$) for P_A requires $\Omega(n)$ queries.*

Consider a set of faulty gates and let A denote the set of their characteristic vectors. Note that, for any faulty parity gate with characteristic vector $\sigma \in \mathcal{F}_2^n$, the output of the gate on input $i \in \mathcal{F}_2^n$ is given by the inner product $\sigma \cdot i$; so, querying the gate q times is equivalent to asking q bits of 2^n-bit x where $x_i = \sigma \cdot i$. Also note that, for any σ, σ', there are exactly $2^n/2$ different i such that $\sigma \cdot i \neq \sigma'_i$. Since any classical algorithm to detect all faults should be able to classify if a given faulty gate belongs to A or not, we get the following lower bound as a corollary to the theorem above.

Corollary 5.4.4 *Any bounded error classical algorithm for detecting all faults of a faulty parity gate requires $\Omega(n)$ queries.*

5.5 Algebraic query complexity

The problem of detecting faulty inputs of gates by querying the gates can be approached from the algebraic query complexity perspective. Usually the nodes of a decision tree is labelled by x_i, the i-th bit of the input x. Generalising it, the detection algorithms can be modelled as decision trees whose nodes are labelled by $g(x, i)$ where g is a fixed Boolean function that takes as input x, the input to the decision tree and i, a value depending on the node.

To relate detecting faulty inputs with algebraic decision trees, consider the problem of detecting faults in a parity gate $\oplus_S$. Any algorithm for such a problem can be viewed

as a decision tree whose input is σ, the characteristic vector for $\mathcal{S}$, and whose nodes are labelled by $g(\sigma, i) = \oplus_k(\sigma_k \cdot i_k)$. Similarly, the nodes in the decision tree for an AND gate $AND_{\mathcal{S}}$ are labelled by $\bigwedge_k(\overline{\sigma_k} \vee i_k)$.

Some of the quantum query complexity results can be shown to also work for quantum algebraic decision trees. For example, Grover's unordered search [Gro96] algorithm can be extended to include algebraic functions at the nodes.

5.5.1 Generalised unordered search

We need a technical result, showing that it is possible to create a unitary transformation which transforms a particular pure state into another pure state. We include the proof for completeness.

Lemma 5.5.1 *Given any two pure states over n qubits, $|x\rangle$ and $|y\rangle$ where $|x\rangle \neq |y\rangle$, there is a unitary transformation U such that $U|x\rangle = |y\rangle$ and $U|y\rangle = |x\rangle$.*

Proof. Let $|b_1\rangle = \frac{1}{\sqrt{2}}|x\rangle - \frac{1}{\sqrt{2}}|y\rangle$ and $|b_2\rangle = \frac{1}{\sqrt{2}}|x\rangle + \frac{1}{\sqrt{2}}|y\rangle$. Note that $|b_1\rangle$ and $|b_2\rangle$ are orthonormal vectors. Extend them to form a complete orthonormal basis for the $N = 2^n$ dimensional Hilbert space $\{|b_1\rangle, |b_2\rangle, \ldots, |b_N\rangle\}$. Observe that, $|x\rangle = \frac{1}{\sqrt{2}}|b_1\rangle + \frac{1}{\sqrt{2}}|b_2\rangle$ and $|y\rangle = -\frac{1}{\sqrt{2}}|b_1\rangle + \frac{1}{\sqrt{2}}|b_2\rangle$.

Then U can be defined as the reflection about the hyperplane orthogonal to $|b_1\rangle$:

$U = -|b_1\rangle\langle b_1| + \sum_{i=2}^{N} |b_i\rangle\langle b_i|$. □

Now we show how to construct a circuit to perform unordered search with generalised oracle gates. The proof closely follows the steps of the original proof by Grover [Gro96].

Theorem 5.5.2 (Unordered search with generalized oracle) *Let $g : \{0,1\}^n \times Y \to \{0,1\}$ be a Boolean function, where the second input to g is from a subset $Y \subseteq \{0,1\}^k$ for some k. Let x be an n-bit binary input, $x = x_1, \ldots, x_n$, and O_x be an oracle gate to compute g, so g: $O_x|i, b\rangle = |i, b \oplus g(x, i)\rangle$.*

Then there is a quantum oracle circuit C, with $O(\sqrt{|Y|})$ oracle gates, such that the measurement of the output qubit of $C|0^n, w\rangle$ (w denotes the oracle workspace qubits) is any $\hat{i} \in Y$ such that $g(x, \hat{i}) = 1$ with constant probability.

Given:

- A subset of k bit binary strings $Y \subseteq \{0,1\}^k$
- n-bit binary input $x \in \{0,1\}^n$
- A function $g : \{0,1\}^n \times Y \to \{0,1\}$ and an oracle gate to compute g: $O_x|i, b\rangle = |i, b \oplus g(x, i)\rangle$

Output: The result of the measurement is any $\hat{i} \in Y$ with probability $O(1)$ such that $g(x, \hat{i}) = 1$.

Proof. As usual, the oracle gate O can be transformed, using one extra ancilla, to change the phase of the register instead of changing the state: $O_x|i\rangle = (-1)^{g(x,i)}|i\rangle$. The ancilla can be reused for all the oracle calls, so we will not explicitly mention it while describing the circuit.

Let $|\Psi\rangle = \sum_{y \in Y} |y\rangle$. We need a unitary operation H' to create a uniform superposition of all elements in Y: $H'|0^n\rangle = |\Psi\rangle$ and $H'|\Psi\rangle = |0^n\rangle$. We know there is gate which performs this operation from lemma 5.5.1. Note that, this gate does not query the oracle gate and so is not counted towards the cost of the circuit.

The circuit performs the following actions in order:

1. Start with $|0^n\rangle$.
2. Apply H' to get $|\Psi\rangle$.
3. Repeat the following in order t number of times.
    - Apply the oracle gate.
    - Apply H'.

- Perform $\Pi = 2|0\rangle\langle 0| - I$ which negates the phase of every basis state except $|0\rangle$. This operation again does not incur any cost.
- Apply H'.

Let $G = (H'(2|0\rangle\langle 0| - I)H')\,O = (2|\Psi\rangle\langle\Psi| - I)O$. Let $Y' = \{i \in Y \mid g(x,i) = 1\}$ denote the solution space and $Y'' = Y \setminus Y'$. Let $|\alpha\rangle = \frac{1}{\sqrt{|Y'|}} \sum_{i \in Y'} |i\rangle$ and $|\beta\rangle = \frac{1}{\sqrt{|Y''|}} \sum_{i \in Y''} |i\rangle$.

The rest of the proof follows as in the original proof [Gro96]. $|\Psi\rangle$ is in the space spanned by $|\alpha\rangle$ and $|\beta\rangle$. O performs a reflection about $|\beta\rangle$ and $(2|\Psi\rangle\langle\Psi| - I)$ performs a reflection about $|\Psi\rangle$, both in the same plane. Thus in effect, G rotates any state in this space towards $|\alpha\rangle$. The final measurement gives us a uniformly chosen element of Y' with probability close to 1 after $O(\sqrt{\frac{|Y|}{|Y'|}})$ queries to the oracle gate. □

There are several subtleties in using the unbounded search technique e.g. for a deterministic running time, the number of solutions should to be known beforehand. There are many variations of the original technique to deal with such issues. All these techniques also work with the generalised oracle.

5.5.2 Lower bound for generalised query complexity

Ambainis developed a lower bound technique for quantum decision trees known as the adversary method [Amb00]. Aaronson later developed a lower bound technique for randomised decision trees using similar ideas [Aar06]. Upon careful examination of the proof, we observe that the main theorems of both the techniques can be extended to work with generalised decision trees. We omit the proofs here which are simple modifications of the original proofs.

Theorem 5.5.3 *[Amb00, Th. 2] Let $g : \{0,1\}^n \times Z \to \{0,1\}$ be a Boolean function, where $Z \subseteq \{0,1\}^k$ for some k. Let O_x be an oracle gate to compute g: $O_x|i,b\rangle = |i, b \oplus g(x,i)\rangle$. Let $f(x_1,\ldots,x_n)$ be a function and $X, Y \subseteq \{0,1\}^n$ be two sets of inputs such that $f(x) \neq f(y)$ if $x \in X, y \in Y$. Let $R \subseteq X \times Y$ such that*

1. $\forall x \in X, \exists^m y \in Y, (x,y) \in R$
2. $\forall y \in Y, \exists^{m'} x \in X, (x,y) \in R$
3. $\forall x \in X, z \in Z$, *there exists at most* l *different* $y \in Y$ *such that* $(x,y) \in R$ *and* $g(x,z) \neq g(y,z)$
4. $\forall y \in Y, z \in Z$, *there exists at most* l' *different* $x \in X$ *such that* $(x,y) \in R$ *and* $g(x,z) \neq g(y,z)$

Then, any oracle quantum circuit computing f *uses* $\Omega(\sqrt{\frac{mm'}{ll'}})$ *queries to* O_x.

Theorem 5.5.4 *[Aar06, Th. 5] For an n-bit Boolean function F, let* $\mathcal{A}$ *denote the set of inputs* $\{x \mid F(x) = 0\}$ *and* $\mathcal{B}$ *denote the set of inputs which evaluate to 1. Let* $g : \{0,1\}^n \times 1, \ldots, n \to \{0,1\}$ *be a Boolean function used for the nodes in the decision tree. Let* $R(A,B) \geq 0$ *be a real-valued function, and for* $A \in \mathcal{A}, B \in \mathcal{B}$ *and* i *(representing the queries that F can make to the input) let,*

$$\begin{aligned} \theta(A,i) &= \frac{\sum_{B^* \in \mathcal{B}: g(A,i) \neq g(B^*,i)} R(A,B^*)}{\sum_{B^* \in \mathcal{B}} R(A,B^*)} \\ \theta(B,i) &= \frac{\sum_{A^* \in \mathcal{A}: g(A^*,i) \neq g(B,i)} R(A^*,B)}{\sum_{A^* \in \mathcal{A}} R(A^*,B)} \end{aligned}$$

where the denominators are all non-zero. Then the number of randomised queries needed to evaluate F with at least 9/10 probability is $\Omega(1/v_{min})$*, where*

$$v_{min} = \max_{\substack{A \in \mathcal{A}, B \in \mathcal{B}, i \\ R(A,B) > 0, g(A,i) \neq g(B,i)}} \min\{\theta(A,i), \theta(B,i)\}$$

5.6 AND gate

We next consider faulty gates computing the AND function $AND_{\mathcal{S}}(x_1, \ldots, x_n) = \bigwedge_{i \notin \mathcal{S}} x_i$. As before, we adopt the convention that $AND_{\{1,2,\ldots,n\}} = 0$. The corresponding quantum gate is basically the unbounded Toffoli gate

$$U|x_1, \ldots, x_n, y\rangle = |x_1, \ldots, x_n, y \oplus \big(AND_{\mathcal{S}}(x_1, \ldots, x_n)\big)\rangle$$

Unlike Parity gate, for this gate, we will show our results for probabilistic algorithms. We will present a $O(\sqrt{n})$-query quantum algorithm to output one of the faults. We will also prove that our quantum detection algorithm makes optimum (up to a constant) number of calls to the faulty gate. We will then show a lower bound of $\Omega(n)$ for classically detecting whether any input of an AND gate is faulty with high probability. Similar to the parity gate, this bound is tight for a classical algorithm: The ith input is faulty if the output of the gate on input f_i is 1.

5.6.1 Quantum detection algorithm

We can use the generalised unordered search from theorem 5.5.2 to detect a faulty input for a Toffoli gate with high probability.

Given $AND_{\mathcal{S}}$, think of x as σ, the characteristic vector of S and take $Y = \{f_1, \ldots, f_n\}$. The action of querying the Toffoli gate can be written as

$$T_{\mathcal{S}}|y_1, \ldots, y_n, b\rangle = |y_1, \ldots, y_n, b \oplus \bigwedge_{i \notin \mathcal{S}} y_i\rangle = |y_1, \ldots, y_n, b \oplus \bigwedge_{i:\sigma_i=0} y_i\rangle$$

which we will use as the oracle gate to compute $g(\sigma, y) = \bigwedge_{i:\sigma_i=0} y_i$ for $y \in Y$.

By theorem 5.5.2, we can construct an oracle quantum circuit that uses $O(\sqrt{n})$ queries to $T_{\mathcal{S}}$ and outputs any f_j such that $\wedge_{i \notin \mathcal{S}}(f_j)_i = 1$ i.e. $j \in S$. Thus the position of the 0 in the output will indicate the position of a faulty input.

Theorem 5.6.1 *For a (possibly) faulty Toffoli gate on n inputs, there is a quantum circuit on n inputs, making $O(\sqrt{n})$ calls to the faulty gate, whose output qubit indicates one of the faulty inputs with high probability.*

Faults in other quantum Boolean gates

Observe that our technique for determining faults in a Toffoli gate can be also used to determine faults for other quantum Boolean gates.

Parity: Though we described a quantum detection circuit for a Parity gate, the above technique can be also be used to detect faults. Instead of the Parity function, we will use its complement, an equivalent function $\bigoplus'(x_1, \ldots, x_n) = 1 \oplus (\bigoplus(x_1, \ldots, x_n))$. The quantum analogue of this gate is the parity gate followed by an X gate on the target qubit, so the faulty lines are preserved in the modified gate.

Take $Y = \{e_1, \ldots, e_n\}$. The action of the faulty gate can be described as, for $y \in Y$,

$$\bigoplus_S |y_1, \ldots, y_n, b\rangle = |y_1, \ldots, y_n, b \oplus \left(\overline{\bigoplus_{i \notin S} y_i}\right)\rangle$$

which is an oracle gate for $g(\sigma, e_i) = 1$ iff $i \in S$.

After $O(\sqrt{n})$ queries, with high probability the output of the circuit is some e_i such that $g(\sigma, e_i) = 1$ i.e. $i \in S$.

5.6.2 Lower bounds

We show tight lower bounds for bounded error detection of faults for both the quantum Toffoli gate and the classical AND gate using the generalised lower bound techniques mentioned in Section 5.5.2.

Let $X = \{00\ldots0\}$ denote a set of gates with no faults and $Y = \{e_1, \ldots, e_n\}$ denote a set of gates with exactly one faulty input. Take $R = X \times Y$. Let $Z = \{f_1, \ldots, f_n\}$. Observe that, we can assume all queries are from $\{f_i\}$. For other queries, the output is always 0 or 1 irrespective of the input. Then the Toffoli gate acts as an oracle gate for the following function: $g(\sigma, f_i) = \bigwedge_{\sigma_j=0}(f_i)_j = 1$ iff i is the only faulty input. For any gate $t \in X \cup Y$, let σ_t denote the characteristic vector of the fault set of that gate.

For $x = 00\ldots0$, there are n elements in Y such that $(x, y) \in R$. On the other hand for any $e_i \in Y$, there is exactly one $x \in X$ such that $(x, y) \in R$. Then in Theorem 5.5.3, $m = n$, $m' = 1$. To compute l, observe that for any f_i, e_i is the only gate such that $g(00\ldots0, f_i) \neq g(\sigma_{e_i}, f_i)$ so $l = 1$. On the other hand, for any $e_i \in Y$, $g(00\ldots0, f) \neq g(\sigma_{e_i}, f)$ only if $f = f_i$; so $l' = 1$. Substituting these values in the theorem, we get the following lower bound for any circuit that can distinguish between X and Y:

Corollary 5.6.2 *Any quantum oracle circuit that can detect if an n-input Toffoli gate has one or no faulty input lines with high probability, makes $\Omega(\sqrt{n})$ queries to the gate.*

For the classical AND gate, we will use Theorem 5.5.4. Using $\mathcal{A} = X, \mathcal{B} = Y$ and using the same relation R as above, we can use similar counting arguments to obtain the following lower bound for randomised fault detection of AND gates.

Corollary 5.6.3 *Any randomised algorithm which detects if an AND gate on n inputs has exactly one or no faulty inputs with high probability requires $\Omega(n)$ calls to the faulty gate.*

5.6.3 Path query model for directed graphs

We would like to point out an interesting correspondence of the fault detection problem with the following graph problem which is of independent interest $\mathbf{FIND_{PATH}}$: Assume we are given a *directed acyclic* graph $G = (V, E)$ on n vertices where there is a self-loop on every vertex. One of the vertices $v^* \in V$ is secretly marked and our goal is to find it. We are allowed to ask if there is a path from *any* vertex to v^*.[4]

It turns out that this problem is related to finding faults of an AND gate. The similarity will be evident if we consider another related problem $\mathbf{FIND_{SUBSET}}$: Given a set S of n elements, to find a hidden subset $s \subseteq S$. We are allowed to query, for any subset x, is $x \subseteq s$? Observe that $\mathbf{FIND_{SUBSET}}$ is a special case of $\mathbf{FIND_{PATH}}$ with G as the n-dimensional hypercube. The vertices of G are labelled by subsets of S and there is a directed edge between from x to y if y is x and exactly one new element. Then, a path exists from x to s in G if and only if $x \subseteq y$.

Next we show that $\mathbf{FIND_{SUBSET}}$ is equivalent to finding faults in an AND gate. A $\mathbf{FIND_{SUBSET}}$ for $S = \{1, \ldots, n\}$ and hidden subset s is equivalent to a faulty n-bit gate AND_s (the ith input is faulty iff $i \in s$). A subset query $x \subseteq s$ is the same as $AND_s(\overline{x})$

[4]Here is a tongue-in-cheek application of this problem. Say, King Arthur wants to promote tourism in his kingdom but does not want to reveal the location of his legendary castle of Camelot[Wika]. Feel free to time travel but keep a foot in 21st century, for his tourism ministry has created an elaborate underground subway by which anyone can travel from any place to any other place, including the castle, yet not know the route or the geographical location. Everybody is otherwise tourist friendly, so maps are readily available showing all the cities in the kingdom. $\mathbf{FIND_{PATH}}$ deals with the problem of finding the city with the castle of Camelot!

where $\overline{x}$ is the complement of the characteristic vector for x ($\overline{x}_i = 0$ iff $i \in x$). This equivalence will be used in the lower bound proof below.

For the general problem of $\mathbf{FIND_{PATH}}$, the marked vertex v^* can be found using $O(n)$ path-queries by a simple algorithm described next. The algorithm first finds any vertex v that has a path to v^*. It then creates a traversal tree for C rooted at v. Finding v^* then amounts to travelling down the tree along any branch that reaches v^*; v^* is the last vertex on that branch to have a path to v^*. The worst case query complexity is also $\Omega(n)$ - consider a star graph and mark one of the $n-1$ edge vertices; any algorithm which is given the graph and asks fewer than $n-2$ queries cannot reliably determine which vertex is marked.

5.7 Conclusion

In this chapter, we considered a fault model where inputs to a Boolean gate might not be connected to the internal gate hardware. We noticed that the Parity gate, which we believe is harder than the AND gate in a complexity theoretic sense, can be diagnosed more efficiently for such faults. We also showed that it is more efficient to detect faults in quantum gates computing these functions than the classical gates.

For both the Boolean functions considered here, Parity and AND, a classical circuit can detect faults in the corresponding classical gates using n queries. For AND, query the gate on $f_1, \ldots, f_n$ successively; if the output on f_i is 1, then the i-th input is faulty. For a Parity gate, query the gate on $e_1, \ldots, e_n$; the output on e_i is 0 if and only if the i-th input is faulty. But it is not clear if there is any non-trivial upper bound to the number of queries for any general Boolean function.

Similarly, we do not know if there is a general lower bound for classical circuits, for either deterministic or probabilistic detection. For example, detecting the faulty input in a Majority gate[5] with only one fault can be done in $\lceil \log_2 n \rceil$ queries. Consider the case

[5] $MAJ_n(x_1, \ldots, x_n) = 1$ iff $\sum_i x_i \geq n/2$

that n is odd (the other case is similar). Observe the output of the gate on $\lfloor n/2 \rfloor$ 0's and $\lceil n/2 \rceil$ 1's. If the output is 1, then the fault is within the inputs which were set to 0 and vice versa. This strategy can be used to binary search for the faulty input. We believe the bound is tight.

Quantum techniques might turn out to be more efficient in determining the sensitivity of a function. Consider the query complexity of distinguishing between $s(f) = 0$ and $s(f) = 1$, given only black box access to f. Whereas $s(f) = 0$ implies f is a constant function, $s(f) = 1$ can be shown to imply that f is balanced (exactly half of the inputs evaluate to 0). While any classical deterministic algorithm must make $\Omega(2^n)$ queries to f, using the well known Deutsch-Jozsa algorithm [DJ92], there is quantum algorithm to distinguish between these two cases using one query to f. However, the technique cannot be generalised to other values of $s(f)$. The query complexity of $s(f)$ is somewhere between $\Omega(n)$ and $O(2^n)$; it would be interesting to close the gap. In addition to it, it would be worthwhile to investigate what additional properties of f would make it easier to determine $s(f)$ [6].

We also considered a graph problem which bears an interesting relation to finding faults of an AND gate. As we showed earlier, the deterministic query complexity for the finding a marked vertex using path queries is $\Theta(n)$; however we also showed that for a hypercube with m vertices, the query complexity is much lower, $\Theta(\log m)$. We would like to ask what, if any, is the relation of the path-query complexity to any structural property of the graph.

There are a several other natural extensions of the topics discussed here which may be worth pursuing. One is to use the techniques examined here to find defects in more complicated circuits, specifically circuits involving several gates and different types of gates. Some of the techniques discussed here are fairly general and might be applicable for other gates as well, e.g. the quantum circuit construction for the faulty Toffoli gate. A second extension is to try to find automatic fault correction strategies for faulty gates and

[6] E.g., can we relate the query complexity of $s(f)$ to $s(f)$ itself ?

circuits. Such strategies often go hand in hand with fault detection in the classical case. We leave the exploration of these ideas for future research.

6 The Last Chapter

" · · · It is wrong to think that the task of physics is to find out how nature is · · · (Niels Bohr)
Us: How are you, Nature?
Nature: Don't ask, I have a terrible headache."
– Frederic Green

In this concluding chapter we take a break from formulating and answering questions and instead summarise the main ideas of this thesis. We also discuss some potential future directions and mention some relevant research areas that were out of scope for discussions in the previous chapters.

6.1 Quantum complexity classes

As mentioned earlier, in *complexity theory*, problems are classified among classes based on their resource usage[1]. The goal is to study the basic properties of the classes instead of the individual problems and organise them in a hierarchical manner. Apropos classical and quantum circuits, the functions computed by them can be similarly classified among circuit classes. Some of the results we described earlier can be understood in terms of the quantum circuit classes. In Table 6.1 below, we list the quantum classes and compare them with their classical counterparts. The classification is for polynomial size circuits

[1]Strictly speaking, a particular problem does not use any resource, but an algorithm for that problem does. For classification, usually the worst-case and sometimes the average-case usage is considered.

and based on the depth of the circuits (the superscript k denotes circuits with depth $O(\log^k n)$) and the types of the gates used therein. Note that fanout is usually taken for granted in a classical circuit, so there is no version of a classical circuit class with fanout gates.

NC^k: *NOT* gates and bounded fanin *AND*,*OR* gates	QNC^k: single qubit and CNOT gates
	$\mathsf{QNC}^k_{\mathrm{wf}}$: single qubit, CNOT and fanout gates
AC^k: *NOT* gates and unbounded fanin *AND*,*OR* gates	QAC^k: single qubit and Toffoli gates
	$\mathsf{QAC}^k_{\mathrm{wf}}$: single qubit, Toffoli and fanout gates
$\mathsf{ACC}^k[q]$: AC^k + MOD_q gates, $\mathsf{ACC}[q] = \mathsf{ACC}^0[q]$	$\mathsf{QACC}^k[q]$: QAC^k + MOD_q gates, $\mathsf{QACC}[q] = \mathsf{QACC}^0[q]$
ACC^k: $\cup_q \mathsf{ACC}^k[q]$, $\mathsf{ACC} = \mathsf{ACC}^0$	QACC^k: $\cup_q \mathsf{QACC}^k[q]$, $\mathsf{QACC} = \mathsf{QACC}^0$
TC^k: AC^k + unbounded fanin threshold gates	QTC^k: QAC^k + arbitrary fanin threshold gates
	$\mathsf{QTC}^k_{\mathrm{wf}}$: $\mathsf{QAC}^k_{\mathrm{wf}}$ + arbitrary fanin threshold gates

Table 6.1: Circuit classes based on the types of the constituent gates

For the classical classes, all we know is $\mathsf{NC}^k \subseteq \mathsf{AC}^k \subseteq \mathsf{ACC}^k \subseteq \mathsf{TC}^k \subseteq \mathsf{NC}^{k+1}$ for any $k \geq 0$ and specifically $\mathsf{NC}^0 \subset \mathsf{AC}^0 \subset \mathsf{ACC}[2]$ ([FSS84]). Also, the Razborov/Smolensky Theorem [Raz87, Smo87] says that for any relatively prime q, p, $\mathsf{ACC}[q] \neq \mathsf{ACC}[p]$, and hence $\mathsf{ACC}[q] \not\subseteq \mathsf{AC}^0$. Nothing much is known about the rest of a huge number of classes.

The number of classes for quantum circuits is even larger. Surprisingly and happily, however, many of them are either the same or are very close, unlike the corresponding classical classes. Since the parity gate and the fanout gate are constant depth-equivalent, it is immediate that $\mathsf{QAC}^0_{\mathrm{wf}} = \mathsf{QACC}_{\mathrm{wf}}[2] = \mathsf{QACC}[2]$. It is also immediate that for all k, $\mathsf{QAC}^k_{\mathrm{wf}} = \mathsf{QACC}^k_{\mathrm{wf}}[2] = \mathsf{QACC}^k[2]$. Since it is possible to fanout n copies in $\log n$ depth using CNOT (via divide and conquer), we also have $\mathsf{QAC}^k \subseteq \mathsf{QAC}^k_{\mathrm{wf}} \subseteq \mathsf{QAC}^{k+1}$. We do not know yet if $\mathsf{AC}^0 \subset \mathsf{QAC}^0$ (the former uses fanout) but for all k, $\mathsf{NC}^k \subseteq \mathsf{QNC}^k_{\mathrm{wf}}$, $\mathsf{AC}^k \subseteq \mathsf{QAC}^k_{\mathrm{wf}}$ and so on. In chapter 3, we explicitly showed how to construct a single circuit for each k that can simulate all circuits in $\mathsf{QAC}^k_{\mathrm{wf}}$ and $\mathsf{QACC}^k_{\mathrm{wf}}$.

Even more interestingly, although the classical $\mathsf{ACC}[m]$ classes could be incomparable as we mentioned above, for quantum classes the opposite turns out to be true:

$\mathsf{QACC}[m] = \mathsf{QACC}[2]$ for all m [GHMP02]. This is exactly the opposite of what we expect from Razborov/Smolensky: $\mathsf{ACC}[p] \neq \mathsf{ACC}[2]$ for any prime power p. Regarding constant-depth bounded fanin circuits, even though they cannot compute much on their own, it turns out that they are very powerful with fanout: Each of the classes $\mathsf{QNC}^0_{\mathrm{wf}}$, $\mathsf{QAC}^0_{\mathrm{wf}}$, or $\mathsf{QTC}^0_{\mathrm{wf}}$ can simulate any of the other two with two-sided polynomially small error [HŠ05], so they roughly possess the same power.

6.2 Analysis of quantum circuits

The quantum circuits that are[2] used in quantum computers may or may not be more complicated than the kinds we considered in the thesis. However, other than showing explicit constructions, another another major goal of this thesis was to analyse these circuits relative to our understanding of classical circuits and try to quantify their efficiencies and deficiencies. We mentioned earlier that QAC^0 is probably weaker than its classical counterpart AC^0 due to the lack of unbounded fanout but other than that we did not emphasise the weakness of a quantum circuit relative to a classical circuit in this thesis.

In the previous chapters we mostly discussed unbounded fanin circuits with single qubit gates and multi-qubit Toffoli and/or parity gates with or without fanout gates. *Structurally*, there are two ways such a circuit is different than an unbounded fanin classical circuit. One is the usage of the single qubit gates which are the primary sources of *entanglement* and creates *superposition*. Observe that the Toffoli gate, the parity gate and the fanout gate are basically classical gates. The other is the restriction of fanout.

Here we would like to point out an interesting resemblance between quantum and classical circuits. Quantum circuits without any ancillæ and fanout behave very much like classical bounded fanin circuits. Without unbounded fanout, the unbounded fanin gates are not capable of quickly mixing up information to generate a complex output. The number of gates in a circuit with no fanout gate is bounded by the product of the

[2] Read: *will be*

depth and the total number of qubits (inputs and ancillæ), so unlike the classical circuits, the number of ancillæ qubits in a quantum circuit is an important resource. It would be interesting to come up with explicit functions which cannot be computed with up to a limited number of ancillæ but can be computed with lots of ancillæ. We would also like to conjecture that for any particular function, a minimum number of gates are required for any quantum circuit to compute it irrespective of the number of ancillæ. The issue of the necessity of ancillæ in quantum computations is a murky one. It is generally accepted that a limited number (polynomially many relative to the number of inputs) are allowed. This seems reasonable as it allows polynomial extra space in which to carry out a computation. However, it is possible to approximate any unitary operator with a small set of universal gates without ancillæ (although one needs circuits of exponential depth and size in order to do so [NC00]). Nielsen addressed this question partly in [Nie06], where he related the minimum size required to exactly implement a specified n-qubit quantum operator without using ancillæ to the length of the minimal geodesic between the operator and the identity operator (length of the geodesic is defined appropriately).

The size of any quantum circuit is upper bounded by the product of its depth and the number of its qubits, including ancillæ. Thus, any bound on the amount of ancillæ required to compute a particular function will also contain insight about the size requirements of that function. A systematic investigation into the absolute necessity of ancillæ for efficient quantum circuits would be an excellent future research direction.

In chapter 2 we surveyed some existing results which show the surprising power provided by allowing fanout gates in constant depth quantum circuits. In the later chapter, we took different approaches to answer the problem of whether, in the presence of single qubit gates, these gates are strictly more powerful than (unbounded) Toffoli gates and are necessary for simulating stronger classes. We seek to fill a gap in our understanding of the relative power of Toffoli and fanout gates in quantum circuits, partly because fanout is a trivial operation for classical circuits. Roughly speaking, we know that given fanout, we can do *fanin*, where by fanin we mean the characteristic property of the quantum gates,

for example the Toffoli gate, in which one output qubit of the gate depends on the values of (unbounded) many inputs qubits. But can fanout do more? Are generalised Toffoli gates and fanout gates equivalent in power, up to polynomial size and constant depth; i.e., are QAC^0 and $\mathsf{QAC}^0_{\mathrm{wf}}$ equal? We believe they are not, and thus that fanout gates are strictly more powerful. The main result of chapter 4 was that one cannot compute parity (and hence fanout) with QAC^0 circuits without using any ancillæ and gave evidence that QAC^0 and $\mathsf{QAC}^0_{\mathrm{wf}}$ may be different.

Polynomial representation A very common technique for analysing classical Boolean formulæ, Boolean circuits and arithmetic circuits is by representing them as polynomials. Unfortunately, there has been no known successful representation of quantum circuits using polynomials to date. One of the difficulty seems to be the presence of amplitudes. The amplitudes contain global information about which states are entangled and how. Conceptually a quantum gate can change the value of the states or amplitudes or both making it difficult for any polynomial to accurately describe the state of only the relevant group of qubits.

Our approach in chapter 4 was motivated by this idea but ended up being insufficient to analyse circuits even with 1 ancilla. There have been some recent attempts in this direction. In [DHH+05], Dawson et. al. represented the probability that of observing, say, 0 in the measurement qubit for a particular input as a polynomial over $\mathcal{Z}_2$. They used that to give another proof that BQP (class of bounded-error problems which could be solved by polynomial size quantum circuits) is contained in the class PP (class of problems solved by probabilistic classical algorithms). They used the *sum-over-paths* approach which can be viewed as the discrete version of the path integral method of standard continuous-time quantum mechanics. This representation does not directly expose the depth of a circuit in a suitable way to allow it to be used directly to answer the questions we discussed in this thesis. However, we believe this approach can be used to analyse circuits with respect to their size.

6.3 Quantum query complexity

In chapter 5, we discussed a specific kind of faulty gates where some of the inputs might not be connected to the gate's output. Our goal was to detect the faulty inputs by using the faulty gate on as few inputs as possible. We considered classical gates computing specific symmetric Boolean functions and also considered quantum gates computing the same. We showed how our fault detection problem is related to determining input sensitivity of Boolean functions. There are several versions of sensitivity e.g. average sensitivity, block sensitivity, which are important complexity measures of functions. The complexity of estimating these values given query access to the function is an interesting research direction which might shed light on the complexity of the sensitivity function itself.

In chapter 5 we also showed how to relate quantum query complexity to quantum circuits. We extended two widely used query complexity results by allowing general algebraic functions for the query steps instead of querying the bits of the input. These results are of independent interest. Algebraic query complexity for quantum circuits would be an interesting area of research. In an algebraic decision tree, we are essentially trying to compute one function with the help of another function. This can help us in understanding the hardness of a function relative to another function e.g. how hard is it to simulate a Toffoli gate using parity gates.

In this thesis we analysed certain types of quantum circuits with an aim to understand what gives them their power beyond classical circuits and in general. We mostly looked at circuits with simple structures that can act as building blocks for complicated circuits. Our focus was both explicit construction and mathematical analysis as and when necessitated. We hope this study will be helpful in understanding complex quantum circuits and quantum computing in general.

ajo nityah sasvato 'yam purano na hanyate hanyamane sarire

He is unborn, eternal, undying, primeval and does not end when the body is slain.

– Srimat Bhagavad Gita 2.20

Bibliography

[Aar06] Scott Aaronson. Lower bounds for local search by quantum arguments. *Siam Journal on Computing*, 35(4):804–824, 2006.

[ADH97] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. Quantum computability. *Siam Journal on Computing*, 26(5):1524–1540, 1997.

[Aha99] Dorit Aharonov. *Noisy Quantum Computation*. PhD thesis, Hebrew University, 1999.

[Aha03] Dorit Aharonov. A simple proof that Toffoli and Hadamard are quantum universal. arxiv:quant-ph/0301040, 2003.

[Amb00] Andris Ambainis. Quantum lower bounds by quantum arguments. In *Proceedings of The 32nd Annual ACM Symposium on Theory of Computing*, pages 636–643. ACM, 2000.

[ASV98] V. Arvind, K.V. Subrahmanyam, and N.V. Vinodchandran. *The Query Qomplexity of Program Checking by Constant-Depth Circuits*, volume 1741/1999 of *Lecture Notes in Computer Science*, pages 123–132. Springer Berlin / Heidelberg, 1998.

[BBC$^+$95] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. Divincenzo, Norman Margolus, Peter W. Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. Elementary gates for quantum computation. *Physical Review A*, A52(5):3457–3487, 1995.

[Bea94] Paul Beame. A switching lemma primer. Technical Report UW-CSE-95-07-01, University of Washington, Computer Science, 1994.

[Bei93] Richard Beigel. The polynomial method in circuit complexity. In *In Proceedings of the 8th IEEE Structure in Complexity Theory Conference*, pages 82–95. IEEE Computer Society Press, 1993.

[Ber08] Debajyoti Bera. A new lower bound technique for quantum circuits without ancillæ. Technical Report 15, Boston University, Computer Science, 2008.

[BFGH09] Debajyoti Bera, Stephen Fenner, Frederic Green, and Steven Homer. Efficient universal quantum circuits. In *Proceedings of The 15th International Computing and Combinatorics Conference*, Lecture Notes in Computer Science. Springer, July 2009.

[BFNR08] Harry Buhrman, Lance Fortnow, Ilan Newman, and Hein Röhrig. Quantum property testing. *Siam Journal on Computing*, 37(5):1387–1400, 2008.

[BGH07] Debajyoti Bera, Frederic Green, and Steven Homer. Small depth quantum circuits. *SIGACT News*, 38(2):35–50, 2007.

[BH09] Debajyoti Bera and Steven Homer. On finding sensitivity of quantum and classical gates. Technical Report 19, Boston University, Computer Science, 2009.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum counting. In *Proceedings of 25th ICALP: Lecture Notes in Computer Science*, volume 1443, pages 820–831. Springer-Verlag, 1998.

[BvDR08] Dave Bacon, Wim van Dam, and Alexander Russell. Analyzing algebraic quantum circuits using exponential sums. http://www.cs.ucsb.edu/~vandam/LeastAction.pdf, 2008.

[BW02] Harry Buhrman and Ronald De Wolf. Complexity measures and decision tree complexity: A survey. *Theoretical Computer Science*, 288(1):21–43, 2002.

[CH85] Stephen A. Cook and H. James Hoover. A depth-universal circuit. *Siam Journal on Computing*, 14(4):833–839, 1985.

[CKW00] Valerie Coffman, Joydip Kundu, and William K. Wootters. Distributed entanglement. *Physical Review A*, 61:052306, 2000.

[Coo04] Matthew Cook. Universality in elementary cellular automata. *Complex Systems*, 15:1–40, 2004.

[Cop94] D. Coppersmith. An approximate fourier transform useful in quantum factoring. Technical Report RC19642, IBM, 1994.

[CTK08] Yao-Hsin Chou, I-Ming Tsai, and Sy-Yen Kuo. Quantum Boolean circuits are 1-testable. *IEEE Transactions on Nanotechnology*, 7(4):484–492, July 2008.

[CW00] Richard Cleve and John Watrous. Fast parallel circuits for the quantum fourier transform. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 526–536, 2000.

[CZ95] J. I. Cirac and P. Zoller. Quantum computations with cold trapped ions. *Physical Review Letters*, 74:4091–4094, May 1995.

[Deu85] David Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. In *Proceedings of the Royal Society of London Ser. A*, volume A400, pages 97–117, 1985.

[DHH+05] Christopher M. Dawson, Henry L. Haselgrove, Andrew P. Hines, Duncan Mortimer, Michael A. Nielsen, and Tobias J. Osborne. Quantum computing and polynomial equations over the finite field Z_2. *Quantum Information and Computation*, 5(2):102–112, 2005.

[DJ92] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Royal Society of London Proceedings Series A*, 439:553–558, October 1992.

[Fen03a] Stephen A. Fenner. A physics-free introduction to the quantum computation model, Computational Complexity Column. *Bulletin of the EATCS*, 79:69–85, 2003.

[Fen03b] Steven A. Fenner. Implementing the fanout gate by a Hamiltonian. arxiv:quant-ph/0309163, 2003.

[Fey82] Richard Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, June 1982.

[FFG+06] Maosen Fang, Stephen Fenner, Frederic Green, Steven Homer, and Yong Zhang. Quantum lower bounds for fanout. *Quantum Information and Computation*, 6(1):46–57, 2006.

[FGHP98] Stephen Fenner, Frederic Green, Steven Homer, and Randall Pruim. Quantum NP is hard for ph. In *Proceedings of 6th Italian Conference on Theoretical Computer Science*, pages 241–252. World Scientific, 1998.

[FR99] Lance Fortnow and John Rogers. Complexity limitations on quantum computation. *Journal of Computer and System Sciences*, 59(2):240–252, 1999.

[FSS84] Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

[GC97] Neil A. Gershenfeld and Isaac L. Chuang. Bulk spin-resonance quantum computation. *Science*, 275(5298):350–356, January 1997.

[GHMP02] Frederic Green, Steven Homer, Cristopher Moore, and Christopher Pollett. Counting, fanout, and the complexity of quantum ACC. *Quantum Information and Computation*, 2(1):35–65, 2002.

[Gre99] Frederic Green. Circuits. Unpublished notes, 1999.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *STOC '96: Proceedings of the twenty-eighth annual ACM Symposium on Theory of Computing*, pages 212–219. ACM, 1996.

[Gro01] Lov K. Grover. From Schrödinger's equation to the quantum search algorithm. *American Journal of Physics*, 69:769–777, 2001.

[Has86] Johan Hastad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th annual ACM Symposium on Theory of Computing*, pages 6–20, 1986.

[Her95] Rolf Herken, editor. *The Universa Turing Machine: A Half-Century Survey*. Springer, 1995.

[HŠ03] Peter Høyer and Robert Špalek. Quantum circuits with unbounded fan-out. In *STACS '03: Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science*, pages 234–246, London, UK, 2003. Springer-Verlag.

[HŠ05] Peter Høyer and Robert Špalek. Quantum fan-out is powerful. *Theory of Computing*, 1(1):81–103, 2005.

[KSV02] A. Yu. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, Boston, MA, USA, 2002.

[MN01] Cristopher Moore and Martin Nilsson. Parallel quantum computation and quantum codes. *Siam Journal on Computing*, 31(3):799–815, 2001.

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

[NC97] Michael A. Nielsen and Isaac L. Chuang. Programmable quantum gate arrays. *Physical Review Letters*, 79(2):321–324, 1997.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, October 2000.

[Nie06] Michael A. Nielsen. A geometric approach to quantum circuit lower bounds. *Quantum Information and Computation*, 6(3):213–262, 2006.

[Nis89] Noam Nisan. CREW PRAMs and decision trees. In *Proceedings of the twenty-first annual ACM Symposium on Theory of Computing*, pages 327–335. ACM, 1989.

[NO02] Harumichi Nishimura and Masanao Ozawa. Computational complexity of uniform quantum circuit families and quantum Turing machines. *Theoretical Computer Science*, 276(1-2):147–181, 2002.

[PBL05] M. Perkowski, J. Biamonte, and M. Lukac. Test generation and fault localization for quantum circuits. In *Proceedings of the 35th International Symposium on Multiple-Valued Logic*, pages 62–68, May 2005.

[PHM04] Ketan N. Patel, John P. Hayes, and Igor L. Markov. Fault testing for reversible circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23:410–416, 2004.

[PP00] S. Parker and M. B. Plenio. Efficient factorization with a single pure qubit and *logn* mixed qubits. *Physical Review Letters*, 85(14):3049–3052, Oct 2000.

[Pud93] P. Pudlák. *Fundamentals of Computation Theory*, chapter AC0 circuit complexity, pages 106 – 120. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 1993.

[Raz87] Alexander Razborov. Lower bounds for the size of circuits of bounded depth with basis $\{\&, \oplus\}$. *Mathematical Notes*, 41(4):333–338, 1987.

[RV01] Vwani P. Roychowdhury and Farrokh Vatan. Quantum formulas: A lower bound and simulation. *Siam Journal on Computing*, 31(2):460–476, 2001.

[RW04] Steven Rudich and Avi Wigderson, editors. *Computational Complexity Theory*, volume 10 of *IAS/Park City Mathematical Series*. American Mathematical Society and Institute of Advanced Study, 2004.

[SBKH93] K.-Y. Siu, J. Bruck, T. Kailath, and T. Hofmeister. Depth efficient neural networks for division and related problems. *IEEE Transactions on Information Theory*, 39(3):946–956, May 1993.

[Shi02] Y. Shi. Both Toffoli and controlled-NOT need little help to do universal quantum computation. *Quantum Information and Computation*, 3(1):84–92, 2002. arXiv:quant-ph/0205115.

[Shi03] Yaoyun Shi. Both toffoli and controlled-NOT need little help to do universal quantum computation. *Quantum Information and Computation*, 3(1):84–92, 2003.

[Sho96] P.W. Shor. Fault-tolerant quantum computation. In *Symposium on Foundations of Computer Science*, volume 0, pages 56–65, Los Alamitos, CA, USA, 1996. IEEE Computer Society.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *Siam Journal on Computing*, 26:1484–1509, 1997.

[Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th annual ACM Symposium on Theory of Computing*, pages 77–82, 1987.

[SR07] Paulo B. M. Sousa and Rubens V. Ramos. Universal quantum circuit for n-qubit quantum gate: A programmable quantum gate. *Quantum Information and Computation*, 7(3):228–242, 2007.

[Tur36] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.

[Val76] Leslie G. Valiant. Universal circuits (preliminary report). In *Proceedings of the 8th ACM Symposium on the Theory of Computing*, pages 196–203, 1976.

[vEB90] Peter van Emde Boas. Machine models and simulations. *Handbook of theoretical computer science (vol. A): algorithms and complexity*, pages 1–66, 1990.

[Vol99] Heribert Vollmer. *Introduction to Circuit Complexity*. Springer-Verlag, 1999.

[Š02] R. Špalek. Quantum circuits with unbounded fan-out. Master's thesis, Faculty of Sciences, Vrije Universiteit, Amsterdam, 2002.

[Wika] Wikipedia. Camelot. *http://en.wikipedia.org/wiki/Camelot*.

[Wikb] Wikipedia. Schrödinger's cat. *http://en.wikipedia.org/wiki/Schrödinger's_cat*.

[Yao77] Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 222–227, 1977.

[Yao93] Andrew C. Yao. Quantum circuit complexity. In *Proceedings of the 34th IEEE Symposium on Foundations of Computer Science*, pages 352–361, 1993.

Curriculum Vitae

Debajyoti Bera

Local address:

111 Cummington Street MCS138, Computer Science Department, Boston, MA 02215, USA

Permanent address:

84B/9 Dumdum Cossipore Road, Kolkata, West Bengal, India - 700074

Email address: dbera@cs.bu.edu

Education

Ph.D. Computer Science, Boston University, USA (expected August 2009).

B.Tech Computer Science, Indian Inst. of Technology, Kanpur, India (2002).

Publications

D. Bera, S. Fenner, F. Green and S. Homer. Efficient Universal Quantum Circuits. Appeared in Proceedings of *COCOON*, 2009.

D. Bera, F. Green, S. Homer. Small depth quantum circuits. SIGACT News, 38(2), 2007.

Work Experience

Internship at ITA Software, Cambridge, USA in summer 2009.

Internship at VMware Inc., Palo Alto, USA in summer 2006.

Developer at Adobe Systems Incorporated, India in 2002.

Internship at GMD IPSI, Darmstadt, Germany in summer 2001.

Mentor for Google Summer of Code for the Beagle project, 2007.

Awards

Best teaching fellow for Computer Sc. Dept. awarded by CAS, Boston Univ. (2006).

Selected for NTSE scholarship, a national merit scholarship by NCERT, India (1996).