

**Quantum Fourier Sampling, the Hidden Subgroup Problem, and Beyond**

by

Sean Joseph Hallgren

B.S. (Carnegie Mellon University) 1994

A dissertation submitted in partial satisfaction of the  
requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION

of the

UNIVERSITY of CALIFORNIA, BERKELEY

Committee in charge:

Professor Umesh V. Vazirani, Chair  
Professor Christos Papadimitriou  
Professor Bernd Sturmfels

Fall 2000

The dissertation of Sean Joseph Hallgren is approved:

---

Chair

Date

---

Date

---

Date

University of California, Berkeley

Fall 2000

# Quantum Fourier Sampling, the Hidden Subgroup Problem, and Beyond

Copyright 2000

by

Sean Joseph Hallgren

## Abstract

Quantum Fourier Sampling, the Hidden Subgroup Problem, and Beyond

by

Sean Joseph Hallgren

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Umesh V. Vazirani, Chair

The Hidden Subgroup Problem (HSP) provides the fundamental framework for most quantum algorithms. Until very recently, all known problems where quantum computation provides a super-polynomial speedup over classical algorithms have been variants of the HSP for particular abelian groups. Examples include factoring integers and computing the discrete log, for which Shor found efficient quantum algorithms. The algorithm for these problems may be viewed as consisting of the main HSP solution plus an ad hoc method of dealing with the particular variant. In this dissertation, we give a systematic way of dealing with all these variants. The key component of our solution is a new theorem about the robustness of Fourier sampling. The purpose of this theorem is to better understand the structure underlying existing algorithms as well as to provide an algorithmic tool for the construction of future quantum algorithms. In addition, we also derive a new algorithm for computing the quantum Fourier transform which is asymptotically faster than

any previously known algorithm.

By contrast, the nonabelian HSP is wide open. It includes as a special case the longstanding open question of graph isomorphism, where the group is the symmetric group,  $S_n$ . It is natural to carry over the abelian HSP algorithm to the nonabelian case. We show that in the case that the hidden subgroup is normal, this algorithm succeeds in reconstructing the subgroup in polynomial time. On the other hand, we give evidence that this algorithm is inadequate to even distinguish a trivial subgroup from an involution in the case of the symmetric group.

Finally, we give an algorithm for the Shifted Legendre Symbol Problem and its variants. There is some evidence that this is an intractable problem classically, and a closely related problem has been proposed as a cryptographic primitive. Perhaps the most interesting aspect of this new quantum algorithm is that it appears to go beyond the framework of the HSP.

---

Professor Umesh V. Vazirani  
Dissertation Committee Chair

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Preliminaries</b>	<b>6</b>
<b>3</b>	<b>The Model</b>	<b>9</b>
3.1	Quantum Algorithms . . . . .	10
<b>4</b>	<b>Fourier Sampling and the HSP</b>	<b>20</b>
4.1	The Hidden Subgroup Problem . . . . .	20
4.2	The General HSP . . . . .	23
4.3	The Nonabelian HSP . . . . .	25
<b>5</b>	<b>Fourier Sampling over Abelian Groups</b>	<b>27</b>
5.1	Introduction . . . . .	27
5.2	Zero-Filling . . . . .	30
5.2.1	The Main Theorem . . . . .	30
5.2.2	Application to the Quantum Case . . . . .	30
5.2.3	The Continuous Picture . . . . .	33
5.2.4	Limitations of Zero-Filling . . . . .	34
5.2.5	Proof of Theorem 5.1 . . . . .	36
5.3	Repeating the Vector . . . . .	39
5.4	Repeating and Zero-Filling . . . . .	41
5.4.1	The Main Theorem . . . . .	41
5.4.2	A Better Fourier Sampling Theorem . . . . .	44
5.4.3	Application: Many-to-One Periodic Functions . . . . .	48
5.4.4	Proof of Theorem 5.3 . . . . .	50
<b>6</b>	<b>Fourier Sampling Coset States of Nonabelian Groups</b>	<b>55</b>
6.1	Representation Theory Background . . . . .	57
6.2	The Probability Distribution over Representations . . . . .	62
6.3	A Positive Result: Normal Subgroups . . . . .	64
6.4	A Negative Result: Determining Triviality in the Symmetric Group . . . . .	66

<b>7</b>	<b>Efficient Quantum Algorithms for Shifted Quadratic Character Problems</b>	<b>69</b>
7.1	Definitions and Related Work . . . . .	70
7.2	An Algorithm for Prime Size Fields . . . . .	72
7.3	An Algorithm for General Finite Fields . . . . .	77
<b>8</b>	<b>Conclusion</b>	<b>81</b>
	<b>Bibliography</b>	<b>83</b>

## Acknowledgements

I would like to thank my advisor Umesh Vazirani for many things, in particular, for patience, and for pointing me in the right directions at the right times. He took on a real project.

I would like to thank my sources of funding. I was supported by an NDSEG Fellowship, GAANN Fellowships, NSF Grant CCR-9800024, and Air Force Grant F30602-00-2-0601.

I would like to thank my parents and Cal Ice Hockey for keeping me sane over the years. My parents for encouraging me along the way, and Cal Ice Hockey for five fun years of games and road trips.

I enjoyed the happy hours, the TGIF's, and the many students I knew. In particular, my officemates and travel companions: Edouard Servan-Schreiber, Sanjoy Dasgupta, Debbie Goldman, Andris Ambainis and Eric Vigoda, along with a long list of others.

I would like to thank my coauthors, especially Lisa Hales, with whom I collaborated with on most of this work and enjoyed working with very much.

Thanks to Peter Høyer and BRICS for their hospitality during my visit there.

I would like to thank Kirsten Eisenträger for reading various versions of this thesis, and for making things fun again.

# Chapter 1

## Introduction

In 1994, Peter Shor [45] made a huge discovery in the field of quantum computing. He discovered that a quantum computer can factor integers efficiently, a problem so strongly believed by researchers to be difficult that all major encryption on the internet is based on the inability to factor integers.

Prior to this there was significant work leading up to Shor's discovery. In 1985 Deutsch [19] defined the first complete model of a quantum computer, but it was not for several years that progress was made in understanding the power of the computational model. Deutsch and Jozsa [20] gave the first nontrivial quantum algorithm. Building upon this Bernstein and Vazirani [8] gave the first example of a problem whose quantum algorithm has a super-polynomial speedup over the best classical algorithm. Their main tool is Fourier sampling, i.e. the process of computing the Fourier transform (of some state), followed by a measurement. Simon [46] then gave a problem and a quantum algorithm that has an exponential speedup over the best classical algorithm. It was from these ideas that Shor

developed his algorithms for factoring and discrete log.

Since Simon's and Shor's discoveries, a framework has been established, through which to understand and extend their results. This framework is built on the Hidden Subgroup Problem (HSP): given a function defined on a finite group, constant and distinct on cosets of an unknown subgroup, find a set of generators for the subgroup. A natural generalization of their algorithms solves the HSP over any abelian group, and this fact has become a folklore theorem over time. Shor's algorithms for factoring and discrete log are somewhat more complicated, and the underlying problem may be abstracted as the variant on the HSP where the group is restricted to be abelian, but with the added complication that the sizes of the cyclic factors of the group are unknown. The solution of this problem follows from a general Fourier sampling theorem we prove in this thesis. A different problem was defined and solved by Kitaev [36], called the Abelian Stabilizer Problem, a problem that has factoring and discrete log as special cases. The stabilizer problem seems very general, but reduces to the HSP (with unknown group size). Kitaev's algorithm provides a completely different point of view of how the HSP algorithm works, but the quantum circuit has been shown to be the same as for the HSP.

Another extremely important question is whether or not a quantum computer can efficiently solve NP-Complete problems, a class of hundreds of problems of great practical importance, and widely believed to be intractable for classical computers. Unfortunately, evidence was provided by Bennett, Bernstein, Brassard, and Vazirani [7] that this class cannot be solved efficiently on a quantum computer, either.

There are at least a few known problems that are probably not NP-Complete,

but appear to be hard classically, and these motivate a search for more efficient quantum algorithms. One example is graph isomorphism, a problem important for practical and theoretical reasons [37]. Another example is the Unique Shortest Lattice Vector Problem, which, like factoring, has cryptography systems [1] based on the fact that no efficient algorithm has been found.

This thesis studies the generalization of Shor’s algorithms to general abelian groups, the generalization of Simon’s algorithm to nonabelian groups, and solves a new problem whose quantum algorithm differs from the HSP framework.

Implicit in Shor’s proof is a robustness of Fourier sampling, for certain restricted initial quantum states, with respect to certain changes in the underlying group. Our first result [31] is a generalization of this: we show that for any quantum state, Fourier sampling is robust under the changes in the underlying group. Underlying our proof is the fact that for a fixed vector, the Fourier transform over  $\mathbb{Z}_p$  can be obtained by considering it as a over  $\mathbb{Z}$ , taking the inverse Fourier transform with respect to the circle group to get a continuous function, and then taking  $p$  evenly spaced points. This solves the principal difficulty in applying the HSP algorithm to solving actual problems where the underlying group is unknown. The robustness theorem may be thought of as providing a compiler where if a problem can be reduced to the HSP where the group is known, then the algorithm will work even if it is not known.

We then combine these ideas with those of [36], [14] and [35] and construct a new Fourier transform algorithm for any abelian group [32]. This algorithm is faster than the previous one [36] and is extremely simple. Due to the simplicity we are able to improve

our Fourier sampling theorem to work on any abelian group, and we get an especially clean explanation of the cyclic case.

By contrast, the nonabelian HSP is wide open. It includes as a special case the longstanding open question of graph isomorphism, where the group is the symmetric group,  $S_n$ . It is natural to carry over the abelian HSP algorithm to the nonabelian case. The algorithm must efficiently compute the Fourier transform, but in the nonabelian case whether this can be done or not must be solved on a group by group basis. For groups we are interested in, such as the symmetric group, it is known [4] how to efficiently compute the quantum Fourier transform. We are mainly interested in how much information can be obtained by Fourier sampling. Ettinger and Høyer [24] showed that the HSP over the dihedral group has polynomial query complexity, but has exponential post-processing time. In this thesis [33] we provide the first classification of how the algorithm for the abelian case of the Hidden Subgroup Problem generalizes for an arbitrary finite nonabelian group. We show that in the case that the hidden subgroup is normal, this algorithm succeeds in reconstructing the subgroup in polynomial time. On the other hand, we give evidence that this algorithm is inadequate to even distinguish a trivial subgroup from an involution in the case of the symmetric group. Independently, Grigni, Schulman, and Vazirani [28] also showed that the algorithm will not work with the reduction, and also showed that an even more general version of the algorithm will not work.

Finally, we give a new quantum algorithm [49] that does not appear to fit into the framework of the HSP. It solves the Shifted Legendre Symbol Problem and its variants. There is some evidence that this is an intractable problem classically, and a closely related

problem has been proposed as a cryptographic primitive. In addition to the fact that the base algorithm is new, there are two other interesting issues that come up. In one variant of the problem a periodic function on a ring is given and the period must be found first. This does not appear to fit into the framework of previous period finding algorithms (namely cyclic subgroups in the HSP problem), and uses our Fourier sampling theorem. The other interesting issue is in the variant defined over finite fields, where the Fourier transform we use must be chosen carefully so that it behaves with respect to both addition and multiplication in the field.

There are other important areas of quantum computation which we do not discuss here. They include quantum error correction, quantum lower bounds, and quantum communication and cryptography. For more on these areas see Nielsen and Chuang [41] or Preskill [43].

Chapter 2 provides a reference to notation of objects used throughout the thesis. Chapter 3 defines the model of a quantum computer. Chapter 4 defines the notion of Fourier sampling, and gives the HSP algorithm. In Chapter 5 we show how to relate Fourier transforms over different groups, resulting in a Fourier sampling theorem and a new Fourier transform algorithm. In Chapter 6 we provide the new results on the nonabelian hidden subgroup problem. Chapter 7 solves the Shifted Legendre Symbol Problem.

## Chapter 2

# Preliminaries

Let  $\mathbb{C}$  denote the field of complex numbers. For  $\alpha \in \mathbb{C}$  let  $\bar{\alpha}$  or  $\alpha^*$  denote the complex conjugate of  $\alpha$ . Let  $\omega^x = e^{2\pi i x}$ . Unit length complex numbers will be written as  $\omega^x$ , where  $x \in [0, 1]$ , since we will want to think in terms of fractions of 1 and want to hide the  $2\pi$ . The principal  $n$ th root of unity  $\omega^{1/n} = e^{2\pi i/n}$  will be written as  $\omega_n$ . Since we will almost always use the notation  $\omega$  instead of  $e^{2\pi i}$ , we will use  $i$  as an index (for example,  $\omega_n^i$ , but not when expanding the  $\omega$  notation into the  $e^{2\pi i}$  form).

All vector spaces will be over the field  $\mathbb{C}$ . Vectors will be written in Dirac's ket notation. In this notation the standard basis of  $\mathbb{C}^p$  for a positive integer  $p$  is  $\{|0\rangle, \dots, |p-1\rangle\}$ . When a vector has a name it is placed inside the ket, for example the vector  $v$  is denoted  $|v\rangle$ . We will also use coefficient names that are the same whenever possible, for example  $|v\rangle = \sum_{i=0}^{p-1} v_i |i\rangle$ ,  $v_i \in \mathbb{C}$ .

An *inner product space* is a vector space  $V$  over  $\mathbb{C}$  with a function  $\langle \cdot, \cdot \rangle : V \times V \rightarrow \mathbb{C}$  satisfying

1.  $\forall |v\rangle \in V, \langle |v\rangle, |v\rangle \rangle \geq 0$ , and  $\langle |v\rangle, |v\rangle \rangle = 0$  iff  $|v\rangle = 0$ .
2.  $\forall |v\rangle, |w\rangle, |x\rangle \in V, \langle \alpha|v\rangle + \beta|w\rangle, |x\rangle \rangle = \alpha\langle |v\rangle, |x\rangle \rangle + \beta\langle |w\rangle, |x\rangle \rangle, \alpha, \beta \in \mathbb{C}$ .
3.  $\forall |v\rangle, |w\rangle \in V, \langle |v\rangle, |w\rangle \rangle = \overline{\langle |w\rangle, |v\rangle \rangle}$ .

A *Hilbert space* is an inner product space  $V$  that is complete with respect to the induced norm  $\| |v\rangle \| = \sqrt{\langle |v\rangle, |v\rangle \rangle}$ , i.e., for any sequence  $\{ |v_n\rangle \}$  with  $|v_n\rangle \in V$ , if  $\lim_{n,m \rightarrow \infty} \| |v_n\rangle - |v_m\rangle \| \rightarrow 0$  then  $\lim_{n \rightarrow \infty} |v_n\rangle \in V$ . Some examples:

- $\mathbb{C}^p$  where  $p$  is a positive integer and  $\langle |v\rangle, |w\rangle \rangle = \sum_{i=0}^{p-1} v_i \overline{w_i}$ .
- $l^2$  is the set of finite norm sequences with  $\langle |v\rangle, |w\rangle \rangle = \sum_{i=-\infty}^{\infty} v_i \overline{w_i}$ , i.e., the set of complex-valued sequences  $\{v_i\}$  such that  $\sum_{i \in \mathbb{Z}} |v_i|^2 < \infty$ .  $l^2$  has basis  $\{|e_i\rangle | i \in \mathbb{Z}\}$ , where  $e_i(i) = 1$  and  $e_i(j) = 0$  if  $j \neq i$ .
- $L^2$  is the set of all finite length functions on  $[0, 1]$  with  $\langle f, g \rangle = \int_0^1 f(x) \overline{g(x)} dx$ , i.e., the set of complex-valued functions on  $[0, 1]$  such that  $\int_0^1 |f(x)|^2 < \infty$ .  $L^2$  has basis  $\{f(x) = \omega^{nx} | n \in \mathbb{Z}\}$ .

The Fourier transform is a norm preserving map on  $\mathbb{C}^p$ . Given a vector  $|v\rangle = \sum_i v_i |i\rangle \in \mathbb{C}^p$ , the Fourier transform is the vector  $|\hat{v}\rangle = \frac{1}{\sqrt{p}} \sum_i \hat{v}_i |i\rangle$ , where  $\hat{v}_i = \sum_j \omega_p^{ij} v_j$ . The Fourier transform is also a distance preserving map from  $L^2$  to  $l^2$ . Given a function  $f \in L^2$ , the Fourier transform of  $f$  is the sequence  $\{\hat{f}(n)\}$ , where  $\hat{f}(n) = \langle f, e_n \rangle = \int_0^1 \omega^{nt} f(t) dt$ . The inverse is  $f(x) = \sum_{n=-\infty}^{\infty} \hat{f}(n) \omega^{xn}$ .

A quantum state with  $n$  qubits is represented by a complex-valued unit vector in  $\mathbb{C}^{2^n}$ , i.e.  $\sum_{i=0}^{2^n-1} v_i |i\rangle$ , where  $\sum_{i=0}^{2^n-1} |v_i|^2 = 1$ . A probability distribution  $\mathcal{D}_{|v\rangle}$  is induced by measuring  $|v\rangle$  and is given by  $\mathcal{D}_{|v\rangle}(i) = |v_i|^2$ .

All groups  $G$  will be finite. If  $H \subseteq G$  is a subgroup and  $c \in G$  is a coset representative, the *coset state* is  $|cH\rangle = \frac{1}{\sqrt{|H|}} \sum_{h \in H} |ch\rangle$ . The Hilbert space in this case is  $\mathbb{C}^{|G|}$ , indexed by elements of the group. Indicator functions of one element or of a set are written as:

- For any  $s$ ,  $\delta_s(x) = \begin{cases} 1 & \text{if } x = s \\ 0 & \text{otherwise} \end{cases}$
- For any set  $S$ ,  $\delta_S(x) = \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise} \end{cases}$

A *character*  $\chi$  of an abelian group  $G$  is a group homomorphism from  $G$  into the multiplicative group of complex numbers of norm 1. The *dual group*  $\widehat{G}$  of an abelian group  $G$  is the set of all characters of  $G$ . The group operation is pointwise multiplication of functions. Given a subgroup  $H \subseteq G$ , the dual of  $G/H$  in  $G$  is  $H^\perp = \widehat{G/H} = \{\chi \in \widehat{G} \mid \chi(h) = 1 \text{ for all } h \in H\}$ .

## Chapter 3

# The Model

In this chapter we will describe the model of a quantum computer, what a quantum algorithm is, and what makes such an algorithm efficient. We will first give a short explanation of how the model is based on quantum mechanics, followed by a more detailed explanation of the model itself.

The model of a quantum computer is based on the (five) postulates of quantum mechanics, which are in turn based on the mathematics of Hilbert spaces. The first postulate says that the state of a system is described by a unit length vector in a Hilbert space. The second says that the evolution of the state is unitary. The third postulate states that the process of measuring a physical quantity, such as energy, position, momentum or angular momentum, has an associated linear Hermitian operator and that the results of the measurement are its eigenvalues. For simplicity we will only use one kind of measurement, measurements in the computational basis. These elements of the model are described in detail in Section 3.1. The model for the quantum computer then adds the assumption that

we can always start in a fixed state, the all zero state, and restricts the unitary operators to be “local”. The latter condition is based on what is thought to be efficiently implementable physically. These restrictions are described in the next section.

### 3.1 Quantum Algorithms

A quantum algorithm starts with  $n$  qubits initially set to zero, evolves them using a unitary transformation, and then a measurement of the bits is performed. The efficiency of the algorithm is measured in terms of the complexity of the unitary transformation performed. We will now describe the setup in detail.

A quantum state is held in  $n$  qubits. A qubit is a unit vector in the Hilbert space  $\mathbb{C}^2$  (with the inner product as in Chapter 2). The space has the standard basis  $\{|0\rangle, |1\rangle\}$  (written in Dirac’s ket notation), so an arbitrary state is written as  $\alpha|0\rangle + \beta|1\rangle$ , where  $\alpha, \beta \in \mathbb{C}$ . This can be thought of as a superposition of the two classical states. A state with  $n$  qubits is a norm one vector in the Hilbert space  $\mathbb{C}^{2^n} = \mathbb{C}^2 \otimes \cdots \otimes \mathbb{C}^2$ , where the tensor product has  $n$  terms. The inner product in this case is defined by  $\langle a \otimes c, b \otimes d \rangle = \langle a, b \rangle \cdot \langle c, d \rangle$ . This space has the standard basis of a tensor product of spaces, which is the set  $\{|b_1\rangle \otimes \cdots \otimes |b_n\rangle \mid b_i \in \{0, 1\}\}$ . We will write the tensor products of two states  $|v\rangle$  and  $|w\rangle$  as  $|v\rangle \otimes |w\rangle$ ,  $|v\rangle|w\rangle$ , or  $|v, w\rangle$ . The set of binary strings of length  $n$  is then written as  $\{|b_1 \cdots b_n\rangle \mid b_i \in \{0, 1\}\}$ . This is referred to as the computational basis. An arbitrary state of this space has the form  $\sum_{i \in \{0,1\}^n} v_i |i\rangle$ , where  $v_i \in \mathbb{C}$  and  $\sum_i |v_i|^2 = 1$ . This can be thought of as a superposition of all possible  $n$  bit classical states.

The second postulate says that evolution is unitary. The notion of complexity

arises from what kind of unitary map is used. A unitary transformation is simple if it operates on two qubits, that is, if it can be written as  $U_{jk} \otimes I$ , where  $U_{jk}$  is an arbitrary unitary operation on bits  $j$  and  $k$ , and the identity is performed on the rest of the bits. The complexity of an operation  $U = U_1 \cdots U_k$ , where each  $U_i$  is simple, is  $k$  quantum steps.

A measurement of the qubits of a state  $\sum_{x \in \{0,1\}^n} v_x |x\rangle$  results in  $x$  with probability  $|\alpha_x|^2$ , and if  $x$  is measured, collapses to the state  $|x\rangle$ . Partial measurements of, say, one qubit, are also possible. In this case the state collapses to the state that is consistent with the measured bit, as shown in the examples below.

Before moving on to the algorithmic part, applying unitary operators, we will first give some examples of states and measurements.

### Example 3.1 (States and Measurements)

- $|0\rangle$  We assume the starting state has all bits zero and that we have as many bits as we want.
- $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$  The Bell state. Suppose we measure the first bit. Then we see a zero with probability  $1/2$  and if we do see zero, the state obtained after measurement is  $|00\rangle$ . Similarly, we see a one with probability  $1/2$  and if we see a one then the state obtained after measurement is  $|11\rangle$ .
- $\frac{1}{\sqrt{3}}(|000\rangle + |001\rangle + |010\rangle)$  If we measure the first bit, we will see a zero with probability  $1$ , and the resulting state is the same. Measuring the second bit results in a zero with probability  $2/3$ , resulting in the state  $\frac{1}{\sqrt{2}}(|000\rangle + |001\rangle)$  if a zero is seen. A one is measured with probability  $1/3$ , resulting in the state  $|010\rangle$  if a one is seen.

■

We will now give some examples of unitary operators applied to various states. The main construction used below is called a controlled- $U$ . The idea is that if we start with a unitary transformation  $U$  that operates on  $n$  bits, and we have an extra bit (or set of bits), we can apply  $U$  conditioned on the other bit. More precisely, if  $|v\rangle$  is a state on  $n$  bits, and  $U$  operates on  $n$  bits, then the following map  $U'$  is a unitary operator and extends linearly:

$$U' : \quad |0\rangle|v\rangle \mapsto |0\rangle|v\rangle$$

$$|1\rangle|v\rangle \mapsto |1\rangle(U|v\rangle).$$

This map is used in the examples below. It may be helpful to notice why each example is reversible.

### Example 3.2 (Unitary Operators on States)

- The controlled-not. Exclusive-ors (addition mod 2) one bit into the next:  $|a, b\rangle \rightarrow |a, a \oplus b\rangle$ . This operation replaces writing a bit to memory in the classical case by keeping it reversible. It is called a controlled-not because it flips the second bit iff the first bit (the control bit) is one.
- Evaluating an oracle. Using the controlled-not we can define an oracle evaluation  $U_f$  by  $|x, y\rangle \xrightarrow{U_f} |x, y \oplus f(x)\rangle$ . Notice that repeating this returns to the original state. We will usually assume  $y = 0$ , and will write  $|x\rangle \rightarrow |x, f(x)\rangle$
- Controlled-rotation. Change the phase of a basis vector  $|p, i\rangle \rightarrow \omega_p^i |p, i\rangle$ , where  $i$  and  $p$  are positive integers. Here the state  $|p, i\rangle$  is the control that specifies what phase

to use. Applied to a superposition we get  $\sum_{i=0}^{p-1} v_i |i\rangle \longrightarrow \sum_{i=0}^{p-1} v_i \omega_p^i |i\rangle$  (when  $p$  is a constant throughout the algorithm, we will not write it down for clarity).

- The Fourier transform  $F_p$ , where  $p$  is a positive integer. Given a basis state  $|c\rangle$ ,  $F_p |c\rangle = \frac{1}{\sqrt{p}} \sum_{i=0}^{p-1} \omega_p^{ic} |i\rangle$ . Measuring all the bits results in a uniformly distributed random number in  $\{0, \dots, p-1\}$ .
- The Fourier transform over any finite group is a unitary operator.

■

We will now describe efficiently implementable unitary transformations. A nice way to describe the allowed operations is to use quantum circuits. Recall that for concreteness, classical algorithms can be described by circuits, using gates like AND, OR, and NOT. Each gate has some wires as inputs and a wire as an output. For example, AND could take two bits  $b_1$  and  $b_2$  and would output  $b_1 \wedge b_2$ . Many gates are connected together so that they compute the output of the algorithm. The input to the algorithm is placed on the inputs to the circuit, the rules of the gates are applied, and the output bits contain the answer. The quantum case is similar, however the gates must be unitary. The condition of unitarity on the quantum gate forces it to have the same number of output wires as input wires. So quantum gates will have some number of inputs and outputs, which will be qubits. The action of the gates can be described by unitary matrices indicating what happens on the input bits. Here are a couple of simple examples.

**Example 3.3** 1. NOT. The NOT gate has one qubit as input and output, and flips its

value. It is described by the  $2 \times 2$  matrix  $N = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ . On the basis states, this performs the maps  $|0\rangle \xrightarrow{N} |1\rangle$  and  $|1\rangle \xrightarrow{N} |0\rangle$ . As is always the case, the operation on arbitrary superpositions can be computed using the linearity of the operation.

2. Controlled-not. A controlled-not has two qubits as inputs and outputs, and performs a NOT on the second bit iff the first bit (the control bit) is one. It is described by the  $4 \times 4$  matrix

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

On the basis states this performs the maps  $|0b\rangle \xrightarrow{C_N} |0b\rangle$  and  $|1b\rangle \xrightarrow{C_N} |1\bar{b}\rangle$ , where  $b \in \{0, 1\}$  and  $\bar{b}$  is not- $b$ .

3. Controlled-rotation. “Rotation” refers to changing the phase of a state, and operates on one bit. Its matrix, for some  $t \in \mathbb{Q}$ , is  $C_{R_t} = \begin{pmatrix} 1 & 0 \\ 0 & \omega^t \end{pmatrix}$ . Recall that  $\omega^t = e^{2\pi it}$ . This performs the maps  $|0\rangle \xrightarrow{C_{R_t}} |0\rangle$  and  $|1\rangle \xrightarrow{C_{R_t}} \omega^t |1\rangle$ .

4. Hadamard transform. The Hadamard transform is a gate with one qubit as input and output. Its matrix is  $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ . On the basis states, this performs the maps  $|0\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  and  $|1\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ . This is also the Fourier transform  $F_2$  over the group  $\mathbb{Z}_2$ .

■

**Example 3.4** Classical computation. Performing a classical computation corresponds to a unitary transformation that is a permutation of the basis states. As the basis states must be permuted, the classical computation must be reversible. All classical computations can be simulated reversibly as long as the input is kept on the tape. Suppose some function  $f$  can be computed efficiently. Then it is possible to compute  $|x\rangle \rightarrow |x, f(x)\rangle$ . If  $f^{-1}$  can be computed, then the computation  $|x, f(x)\rangle \rightarrow |0, f(x)\rangle$  is also possible. As an example it may be helpful to think of the function multiply:  $|p, q\rangle \rightarrow |p, q, pq\rangle$ . To erase  $p$  and  $q$  it must be possible to efficiently compute them from  $pq$ . It is only known how to do this using the quantum factoring algorithm. ■

A polynomial number of gates are put together in a sequence to compute the output of the algorithm. The set of basic gates that we will allow are all one bit gates (i.e., all  $2 \times 2$  unitary matrices), and the controlled-not. This set allows any polynomial time quantum computation to be described by a polynomial size circuit.

We will not describe operations by matrices, since they can get very large, and which bits they operate on must be specified in some way. Instead we will continue to use the other notation  $|v\rangle \xrightarrow{U} U|v\rangle$ . We will give some more examples of efficient operations, and any sequence of at most a polynomial in them will also be efficient.

Notice that because the gates can act on at most two bits at a time, computation is local in some sense. One can imagine being at one of the computational basis states  $|i\rangle$  and being allowed to perform an operation of the type “if the first bit is one, then do..., otherwise do nothing.” We will now give some more examples of operations that can be efficiently implemented.

**Example 3.5** Controlled-rotation. Here we have a more complicated version than before. It is possible to implement the map  $|x\rangle \rightarrow \omega^{x/p}|x\rangle$ , where  $x$  and  $p$  are positive integers. Note that this would require a matrix much larger than the previous example, because it conditions on all  $n$  bits instead of just two. ■

**Example 3.6** Controlled- $U$ . Fix some value  $y$ , and define the map  $|y\rangle \rightarrow U|y\rangle$ , and  $|x\rangle \rightarrow |x\rangle$  if  $x \neq y$ . ■

**Example 3.7** The Fourier transform over the group  $\mathbb{Z}_2^n$ . As mentioned before, the Fourier transform over any group is a unitary operator. We want to describe it in terms of a small circuit. Since  $F_{\mathbb{Z}_2^n} = \bigotimes_{i=0}^{n-1} F_2$ , we can refer to the previous example, and just apply the Hadamard transform to each bit. Given a basis state  $|i\rangle = |i_1 \cdots i_n\rangle$ , where the  $i_j$  are the bits of  $i$ , we have

$$|i\rangle = \bigotimes_{j=0}^{n-1} |i_j\rangle \xrightarrow{\bigotimes F_2} \bigotimes_{j=0}^{n-1} (|0\rangle + (-1)^{i_j}|1\rangle) = \sum_{c=0}^{2^n-1} (-1)^{i \cdot c} |c\rangle,$$

where  $i \cdot c$  is the inner product mod 2 of the bits of  $i$  and  $c$ . ■

A more complicated, but still relatively simple example, is the Fourier transform over  $\mathbb{Z}_{2^n}$ . The algorithm is as easy to understand as it is in the classical case [17]. The most confusing thing is to index everything properly, also as it is in the classical case. The quantum algorithm only takes about  $(\log n)^2$  steps, unlike the classical FFT, which takes  $O(n \log n)$ . The advantage a quantum computer has is that in the recursion step, it can simultaneously compute all the recursive calls at once.

**Example 3.8** The Fourier transform  $F_{2^n}$ . The ability to compute this kind of function is exactly why quantum computation gets a speedup over classical computation. Instead

of listing the algorithm, we will try to illustrate why the algorithm is so much faster by showing how it differs from the classical algorithm.

We will first describe the classical algorithm, taken from [17]. We will use ket notation for the vectors for ease of translation. Assume the algorithm is called on  $|\alpha\rangle = \sum_{i=0}^{p-1} \alpha_i |i\rangle$ , where  $p$  is a power of 2. Similarly,  $|\beta\rangle$  will be the vector returned. Before listing the code we first illustrate how the recursion step works in both cases.

First rewrite the vector as follows, separating the even and odd coefficients, and then recursively compute the Fourier transform:

$$\begin{array}{ccc}
 \left( \begin{array}{c} \alpha_0 \\ \alpha_2 \\ \alpha_4 \\ \vdots \\ \alpha_{p-2} \end{array} \right) & \xrightarrow{F_{p/2}} & \left( \begin{array}{c} \hat{\alpha}_0 \\ \hat{\alpha}_2 \\ \hat{\alpha}_4 \\ \vdots \\ \hat{\alpha}_{p-2} \end{array} \right) \\
 \\
 \left( \begin{array}{c} \alpha_1 \\ \alpha_3 \\ \alpha_5 \\ \vdots \\ \alpha_{p-1} \end{array} \right) & \xrightarrow{F_{p/2}} & \left( \begin{array}{c} \tilde{\alpha}_1 \\ \tilde{\alpha}_3 \\ \tilde{\alpha}_5 \\ \vdots \\ \tilde{\alpha}_{p-1} \end{array} \right)
 \end{array}$$

Now we use the results from the recursion step to form the answer:

$$\begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p/2} \\ \beta_{p/2+1} \\ \vdots \\ \beta_{p-1} \end{pmatrix} = \begin{pmatrix} \hat{\alpha}_0 \\ \hat{\alpha}_2 \\ \vdots \\ \hat{\alpha}_{p-2} \end{pmatrix} + \begin{pmatrix} \omega_p^0 \tilde{\alpha}_1 \\ \omega_p^1 \tilde{\alpha}_3 \\ \vdots \\ \omega_p^{p/2-1} \tilde{\alpha}_{p-1} \end{pmatrix} - \begin{pmatrix} \omega_p^0 \tilde{\alpha}_1 \\ \omega_p^1 \tilde{\alpha}_3 \\ \vdots \\ \omega_p^{p/2-1} \tilde{\alpha}_{p-1} \end{pmatrix}$$

**Algorithm 3.1 (Recursive-FFT( $|\alpha\rangle, p$ ) (assume  $p$  is the length of  $|\alpha\rangle$ ))**

1. if  $p = 1$ , return  $|\alpha\rangle$

2.  $t := 0$

3.  $\sum_{i=0}^{p/2-1} \hat{\alpha}_{2i}|2i\rangle := \text{Recursive-FFT}(\sum_{i=0}^{p/2-1} \alpha_{2i}|i\rangle)$  ; Recurse on even coefficients

4.  $\sum_{i=0}^{p/2-1} \tilde{\alpha}_{2i+1}|2i+1\rangle := \text{Recursive-FFT}(\sum_{i=0}^{p/2-1} \alpha_{2i+1}|i\rangle)$  ; Recurse on odd coefficients

5. for  $k = 0$  to  $p/2 - 1$

(a)  $\beta_k := \hat{\alpha}_{2k} + \omega_p^t \tilde{\alpha}_{2k+1}$

(b)  $\beta_{k+p/2} := \hat{\alpha}_{2k} - \omega_p^t \tilde{\alpha}_{2k+1}$

(c)  $t := t + 1$

6. return  $|\beta\rangle$

The recursion tree has depth  $\log p$  and width  $p$ . This algorithm should be compared with the following, which is the recursive step in the quantum algorithm.

$$\sum_{i=0}^{p-1} \alpha_i |i\rangle = \sum_{i=0}^{p/2-1} \alpha_{2i} |i\rangle |0\rangle + \sum_{i=0}^{p/2-1} \alpha_{2i+1} |i\rangle |1\rangle \quad (3.1)$$

$$\xrightarrow{F_{p/2}} \sum_{i=0}^{p/2-1} \hat{\alpha}_{2i} |i\rangle |0\rangle + \sum_{i=0}^{p/2-1} \tilde{\alpha}_{2i+1} |i\rangle |1\rangle \quad (3.2)$$

$$\xrightarrow{C-R(\omega_p^t)} \sum_{i=0}^{p/2-1} \hat{\alpha}_{2i} |i\rangle |0\rangle + \sum_{i=0}^{p/2-1} \omega_p^t \tilde{\alpha}_{2i+1} |i\rangle |1\rangle \quad (3.3)$$

$$\xrightarrow{F_2} \sum_{i=0}^{p/2-1} (\hat{\alpha}_{2i} + \omega_p^t \tilde{\alpha}_{2i+1}) |i\rangle |0\rangle + \sum_{i=0}^{p/2-1} (\hat{\alpha}_{2i} - \omega_p^t \tilde{\alpha}_{2i+1}) |i\rangle |1\rangle \quad (3.4)$$

In (3.1) the equation is rewritten to emphasize that the vector lies in two different subspaces. In (3.2) the Fourier transform is recursively computed on the the first  $n - 1$  bits. Notice that this simultaneously computes the Fourier transform on both subspaces at once! In (3.3) the phase is added, and in (3.4) the results from the two subspaces are combined. One more minor step must be taken that we did not write, which is to reorder the bits by making the least significant bit the most significant bit. The recursion tree has depth  $\log p$ , but only has width one. ■

**Example 3.9**  $F_p$ , where  $p$  is a prime. It was first shown how to  $\epsilon$ -approximate this in [36].

In Chapter 5 we will give a faster and simpler algorithm. ■

The Fourier transform over an arbitrary abelian group can be computed efficiently since it is a tensor product of the Fourier transform over the cyclic factors.

## Chapter 4

# Fourier Sampling and the HSP

In this chapter we will discuss the problems that can be efficiently solved by quantum computers but cannot be solved efficiently by classical probabilistic algorithms. The main primitive is Fourier sampling, which is the process of computing the Fourier transform over a group  $G$  and measuring. It will be the main step in the algorithm below. Along with providing background, this chapter will motivate the contributions of this thesis, which start in the next chapter.

### 4.1 The Hidden Subgroup Problem

**Definition 4.1 (The Hidden Subgroup Problem (HSP))** *Given a finite group  $G$ , and a function  $f : G \rightarrow S$  that is constant and distinct on cosets of some unknown subgroup  $H$  of  $G$ , where  $S$  is any set. Find a set of generators for  $H$ .*

The main transformation in the algorithm for the HSP is the quantum Fourier transform, which can be efficiently computed over any abelian group. The Fourier transform

is a change of basis of  $\mathbb{C}^{|G|}$  from the basis indexed by group elements  $\{|g\rangle : g \in G\}$ , to the basis indexed by the characters of the group  $\{|\chi\rangle : \chi \in \hat{G}\}$ . The group of characters is the group  $\hat{G} = \{\chi : G \rightarrow \mathbb{C}^*\}$ , where  $|\chi\rangle = \sum_{g \in G} \chi(g)|g\rangle$ . The algorithm for solving the HSP uses two known properties [47] of the Fourier transform over  $G$ . The first is that a state that is uniform on a subgroup  $H$  is mapped to the perp subgroup  $H^\perp = \{\chi : H \subseteq \ker \chi\}$ . That is, we have the map

$$\frac{1}{\sqrt{|H|}} \sum_{h \in H} |h\rangle \xrightarrow{F_G} \frac{1}{\sqrt{|H^\perp|}} \sum_{\chi \in H^\perp} |\chi\rangle.$$

Fourier sampling a subgroup state would give a uniform distribution over the perp group, from which we could compute  $H$ . The second property is that the Fourier transform of the convolution “ $*$ ” of two vectors is the pointwise product “ $\bullet$ ” of the Fourier transform of each vector. Strictly speaking, this is only true without the normalization factors, which we will now leave out. Consider a coset state  $\sum_{h \in H} |gh\rangle$ :

$$\sum_{h \in H} |gh\rangle = |g\rangle * \sum_{h \in H} |h\rangle \xrightarrow{F_G} \left( \sum_{\chi \in \hat{G}} \chi(g) |\chi\rangle \right) \bullet \left( \sum_{\chi \in H^\perp} |\chi\rangle \right) = \sum_{\chi \in H^\perp} \chi(g) |\chi\rangle.$$

Therefore, Fourier sampling treats all coset states the same, i.e. the identity of the coset does not matter. We can now describe the algorithm. In the algorithm  $M_i$  denotes a measurement of the  $i^{\text{th}}$  register. In particular,  $M_2$  measures the value of  $f(g)$ , and  $M_1$  measures the character name.

#### Algorithm 4.1 (Abelian Hidden Subgroup Problem Algorithm)

**Input:** A group  $G$ , a function  $f$  that is constant and distinct on cosets of unknown  $H$ .

**Output:** A set of generators for  $H$ .

1. Repeat  $k = \text{poly}(\log |G|)$  times:

(a) Create a random coset state  $\sum_{h \in H} |gh\rangle$ :

$$|0\rangle \xrightarrow{F_G} \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle \xrightarrow{U_f} \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g, f(g)\rangle \xrightarrow{M_2} \frac{1}{\sqrt{|H|}} \sum_{h \in H} |gh, f(g)\rangle$$

(b) Fourier sample the coset state:

$$\frac{1}{\sqrt{|H|}} \sum_{h \in H} |gh\rangle \xrightarrow{F_G} \frac{1}{\sqrt{|H^\perp|}} \sum_{\chi \in H^\perp} \chi(g) |\chi\rangle \xrightarrow{M_1} |\chi_i\rangle$$

2. Classically compute  $H = \bigcap_{i=0}^{k-1} \ker \chi_i$

Since we are sampling from a uniform distribution on the subgroup  $H^\perp$ , a polynomial number of samples is enough to know  $H^\perp$ , and then the intersection of the kernels can be efficiently computed from the set of modular linear equations they define.

Simon [46] gave the first example of such a problem. The instance given is  $G = \mathbb{Z}_2^n$ , and a function satisfying the HSP, and the problem is to decide whether the hidden subgroup is trivial or has order two.

One problem that is not directly an instance of the HSP, but still uses the same properties, is the recursive Fourier sampling problem of Bernstein and Vazirani [8]. The recursive Fourier sampling problem is the only known example of a problem that is not in NP that can be efficiently solved by a quantum computer. In one level of the recursion, a function  $f_s : \mathbb{Z}_2^n \rightarrow \{0, 1\}$  is given such that  $f_s(x) = (x, s) \bmod 2$ , the inner product mod 2 of  $x$  and  $s$ . The goal is to find  $s$ . The quantum algorithm is to Fourier sample the state with  $f$  in the phases:

$$\sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle \xrightarrow{F_{\mathbb{Z}_2^n}} |s\rangle.$$

Notice that this step can be viewed as the HSP algorithm in reverse, where  $|s\rangle$  is the coset state and the subgroup is trivial.

## 4.2 The General HSP

In the General HSP, less information is given. The input in this case is a function  $f : \mathbb{Z}^k \rightarrow S$  and integers  $q_1, \dots, q_k$ . The guarantee is that the hidden subgroup is contained in a finite size group contained in  $\mathbb{Z}^k$ , and that the values  $q_i$  are estimates of the size of the group in each component. We cannot immediately apply the HSP algorithm in this case since we do not know the group.

This variant arises in Shor's algorithm for factoring. In the algorithm, the idea is to reduce factoring to the HSP over a cyclic group, but in this case the group size is necessarily unknown. Using ad hoc techniques, he showed that it is possible to Fourier sample over a group of a different size and still reconstruct the subgroup. We will now outline the algorithm: given an odd integer  $N$  to factor, suppose for the moment that we can find the order of  $x \in \mathbb{Z}_N$ , i.e., the least  $r$  such that  $x^r \equiv 1 \pmod{N}$ . The following algorithm factors  $N$ . Choose a random  $x \in \mathbb{Z}_N$ , and find its order  $r$ . If  $r$  is even then compute  $\gcd(x^{r/2} - 1, N)$ , otherwise try a new  $x$ . Since with probability at least  $1/2$ , the order  $r$  is even, and  $x^{r/2} \not\equiv \pm 1 \pmod{N}$  [38], we get a factor of  $N$ .

We will now show how to find  $r$ . Note that the order of  $x \pmod{N}$  divides  $\phi(N)$ , where  $\phi(N)$  is Euler's phi function, since the exponent of  $x$  acts as addition mod  $\phi(N)$ . Define the function  $f : \mathbb{Z}_{\phi(N)} \rightarrow \mathbb{Z}_N$  by  $f(x) = x^r \pmod{N}$ . This is an instance of the HSP. The problem is that  $\phi(N)$  is unknown, and indeed it could not be known since the factors of  $N$  can be computed in polynomial time from knowledge of  $\phi(N)$ . Shor showed that nevertheless Fourier sampling over a larger size group  $\mathbb{Z}_q$ , where  $q > \phi(N)$ , results in a distribution similar to the original, and  $r$  can be found using continued fractions.

In Chapter 5 we give new results generalizing Shor's ideas to arbitrary quantum states and arbitrary abelian groups. In Chapter 5 we characterize the behavior of the Fourier transform over different cyclic groups for arbitrary input vectors. The conclusion is that this process is possible in general for abelian groups whose decomposition into cyclic groups has at most a constant number of such factors. We also extend these results to all abelian groups. We first show that repeating the input superposition, combined with computing the Fourier transform over a larger modulus, leads to a new Fourier transform algorithm over abelian groups. The algorithm is asymptotically faster than the previous one, and is extremely simple. Due to the simplicity of the algorithm, we can conclude a new Fourier sampling theorem that works for all abelian groups.

Boneh and Lipton [11] solve a variant of the HSP in certain cases where the function is not distinct on all cosets. We solve this in general, and give necessary and sufficient conditions describing when this is possible.

Kitaev [36] defines the Abelian Stabilizer problem, which seems quite general. It is defined as follows: given positive integers  $k$  and  $n$ , an element  $a \in \{0, 1\}^n$ , and a blackbox  $F$  defining an action of  $G = \mathbb{Z}^k$  on the set  $M \subseteq \{0, 1\}^n$ , find the stabilizer of  $a$ . The stabilizer of  $a$  is the subset of  $G$  that fixes  $a$  under the action, and is a subgroup of  $G$ . This problem includes factoring and discrete log, and if the group is nonabelian, then it also includes graph isomorphism. It is not clear if the problem is equivalent to the HSP, but it does reduce to the General HSP, by defining the function  $f$  by  $f(g) = F(g, a)$ . Then  $f$  is an instance of the hidden subgroup problem, the hidden subgroup being the stabilizer.

There is another approach to efficient quantum algorithms that involves eigenvalue

estimation [36]. However, a link has been found between that method and the HSP algorithm presented here [14], and there are no known problems that it can solve that cannot be solved using Fourier sampling.

### 4.3 The Nonabelian HSP

Ettinger and Hoyer [24] give an algorithm for the HSP for the dihedral group. Their algorithm is close to the HSP algorithm, except they Fourier sample over a different group. While the dihedral group is the semi-direct product  $\mathbb{Z}_N \rtimes \mathbb{Z}_2$ , their algorithm Fourier samples over the abelian group  $\mathbb{Z}_N \times \mathbb{Z}_2$ . They show that a polynomial number of queries (i.e. samples in the HSP algorithm) is enough to reconstruct the hidden subgroup, with exponential classical post-processing time.

In [23, 22, 25], they address the question of whether or not any algorithm is possible at all. They show that the tensor products of coset states for different subgroups are almost orthogonal. This shows a polynomial query complexity, but requires an exponential number of samples (quantum measurements) from the quantum step.

Rötteler and Beth [44] give an example of a nonabelian group where the HSP can be solved. Zalka [51] has shown how to solve the problem using the HSP algorithm over  $\mathbb{Z}_2^b$ .

In this thesis we give the first characterization of the distribution sampled in the HSP algorithm for nonabelian groups. We use this to show that for normal subgroups the distribution has a particularly nice form, and that a polynomial number of samples is enough to reconstruct hidden normal subgroups. We then analyze the reduction of graph isomorphism to the HSP over  $S_n$  and show that the HSP algorithm does not work, by

showing that the trivial subgroup cannot be distinguished from order two subgroups.

Our result about  $S_n$  is based on only sampling the group representation  $\rho$ , and not the rows and columns of the matrices (group representations are homomorphisms into matrix groups). Independently, Grigni, Schulman, and Vazirani [28] have shown that this approach does not work, and in addition show that measuring the row of the matrix in addition does not work.

## Chapter 5

# Fourier Sampling over Abelian Groups

### 5.1 Introduction

In this chapter we analyze the relationship between the Fourier transform over the cyclic group  $\mathbb{Z}_p$  and the Fourier transform over the cyclic group  $\mathbb{Z}_q$ , where  $q > p$ . Since any abelian group is a direct product of cyclic groups, there is a natural extension of the results to general abelian groups. The two main results are a Fourier sampling theorem, showing that Fourier sampling is robust under group change, and a new quantum Fourier transform algorithm that is the most efficient known to date.

The Fourier transform over the group  $\mathbb{Z}_p$  is a map from  $\mathbb{C}^p$  to  $\mathbb{C}^p$ . Recall that Fourier sampling is a map from  $\mathbb{C}^p$  to  $\mathbb{R}^p$ , where given a state  $|v\rangle$ , we get a probability distribution  $\mathcal{D}_{|v\rangle}$ . We wish to study how the Fourier transform and how Fourier sampling,

with respect to  $\mathbb{Z}_q$ , carried out on the state  $|v\rangle$ , relates to the respective operations over  $\mathbb{Z}_p$ . To do this we must first say how to map the state  $|v\rangle$  over  $\mathbb{Z}_p$  to a state over  $\mathbb{Z}_q$ . We must also specify how to map back from  $\mathbb{Z}_q$  to  $\mathbb{Z}_p$  after the Fourier transform. Below we define several maps for each direction. The choice of map is quite crucial in determining how closely the Fourier transforms approximate each other.

We define two maps on the state  $|v\rangle$  from  $\mathbb{Z}_p$  to  $\mathbb{Z}_q$ . The first is the inclusion map  $\iota : \mathbb{C}^p \rightarrow \mathbb{C}^q$  which just identifies  $\mathbb{C}^p$  with a subspace of  $\mathbb{C}^q$  and maps  $|v\rangle$  to itself. The second map  $R_q$  repeats the vector to have  $q$  coefficients. On a vector  $|v\rangle = \sum_{i=0}^{p-1} v_i |i\rangle \in \mathbb{C}^p$ ,  $R_q |v\rangle = c \cdot \sum_{i=0}^{q-1} v_{i \bmod p} |i\rangle$ , where  $c$  is the appropriate normalization factor.

After we compute the Fourier transform over  $\mathbb{Z}_q$ , we have three possible ways of mapping back to  $\mathbb{Z}_p$  to approximate Fourier sampling. A natural map is to divide  $[0, q-1]$  into  $p$  equal intervals, and map all integer points in the  $j^{\text{th}}$  interval to  $j$ . This gives us the bucket distribution  $\mathcal{D}_{|\hat{u}\rangle}^B(i) = \sum_{t \in T} |\hat{u}_{s_i+t}|^2$ , where  $T = \{-\lfloor \frac{q}{2p} \rfloor + 1, \dots, \lfloor \frac{q}{2p} \rfloor\}$ . Another possibility is to use only the midpoint of each interval (i.e. the closest integer to the midpoint). First we define the map  $s(i) = \lfloor \frac{q}{p} i \rfloor$ . Let  $s_i = s(i)$ . The inverse of this map, the projection map  $P : \mathbb{C}^q \rightarrow \mathbb{C}^p$ , is then defined as

$$P|j\rangle = \begin{cases} |i\rangle & \text{if } s_i = j \text{ for some } i \in \{0, \dots, p-1\} \\ 0 & \text{otherwise} \end{cases}$$

Let  $|\hat{u}\rangle$  be the result of the Fourier transform over  $\mathbb{Z}_q$ . The midpoint distribution  $\mathcal{D}^M$  is defined by  $\mathcal{D}_{|\hat{u}\rangle}^M(i) = c^{-2} \cdot \mathcal{D}_{|\hat{u}\rangle}(s_i)$  where the normalization factor is  $c^2 = \sum_{i=0}^{p-1} \mathcal{D}_{|\hat{u}\rangle}(s_i)$ .

There are cases where we must Fourier sample and  $p$  is unknown, but an upper-bound  $n$  is known. In this case, we define the map from  $\mathbb{Z}_q$  to  $\mathbb{Q}$ , such that  $i$  is mapped to the best rational approximation of  $i/q$  with denominator at most  $n$  (this is found by continued

fractions). This induced distribution  $\mathcal{D}^{\text{CF}}$ , is called the continued fractions distribution.

In Section 5.2 we show that by zero-filling and using the midpoint distribution  $\mathcal{D}^{\text{M}}$ , we can approximate Fourier sampling over  $\mathbb{Z}_p$ . The intuitive reason that this works is as follows: the Fourier transform over  $\mathbb{Z}_p$  can be obtained by regarding  $|v\rangle$  as a vector over  $\mathbb{Z}$ , taking the inverse Fourier transform with respect to the circle group to get a continuous function, and then restricting to its values at  $p$  evenly spaced points. Using the midpoint distribution is wasteful in terms of sample complexity, and it is natural to try to use the bucket distribution. However, we give a counterexample showing that using buckets does not even work for cyclic groups.

In Section 5.3 we show that repeating the vector via  $R_q$  and using the bucket distribution gives extremely good approximations to Fourier sampling over  $\mathbb{Z}_p$ . This is based on some results discovered independently in the context of eigenvalue estimation. Since this technique is more efficient in its use of samples, it extends to abelian groups.

In Section 5.4 we show that by combining the two methods of repeating the vector and then zero-filling, and adding another ingredient to map back to  $\mathbb{Z}_p$ , we get a very fast quantum algorithm for the Fourier transform over  $\mathbb{Z}_p$ .

## 5.2 Zero-Filling

### 5.2.1 The Main Theorem

The main theorem of this section may be regarded as showing that the diagram

$$\begin{array}{ccc} \mathbb{C}^p & \xrightarrow{F_p} & \mathbb{C}^p \\ \downarrow \iota & & \uparrow \sqrt{\frac{q}{p}} P \\ \mathbb{C}^q & \xrightarrow{F_q} & \mathbb{C}^q \end{array}$$

“commutes approximately” when  $q$  is sufficiently large compared to  $p$ . By “commutes approximately” we mean that for all  $v \in \mathbb{C}^p$ ,  $|F_p|v\rangle - \sqrt{\frac{q}{p}} P F_q \iota|v\rangle| \leq \epsilon$ . In the context of quantum computation, this allows us to Fourier sample over  $\mathbb{Z}_q$  instead of  $\mathbb{Z}_p$ .

**Theorem 5.1** *For all  $s \geq 0$  and  $|v\rangle \in \mathbb{C}^p$ , if  $q \geq (24 \cdot s \cdot \ln p) \cdot p$ , then*

$$\|F_p|v\rangle - \sqrt{\frac{q}{p}} P F_q \iota|v\rangle\| \leq \frac{\|v\|}{8s}$$

The setup is illustrated with an example in Figure 5.1.

### 5.2.2 Application to the Quantum Case

We will now apply Theorem 5.1 to the quantum case to prove that Fourier sampling over different group sizes works. Let  $|v\rangle$  be a quantum state,  $|\hat{v}\rangle = F_p|v\rangle$ , and  $|\hat{u}\rangle = F_q \iota|v\rangle$ . Let  $\mathcal{D}_{|\hat{v}\rangle} : \{0, \dots, p-1\} \rightarrow [0, 1)$  and  $\mathcal{D}_{|\hat{u}\rangle} : \{0, \dots, q-1\} \rightarrow [0, 1)$  be the probability distributions induced by measuring  $|\hat{v}\rangle$  and  $|\hat{u}\rangle$ , respectively. Let  $\mathcal{D}_{|\hat{u}\rangle}^M : \{0, \dots, p-1\} \rightarrow [0, 1)$  be the distribution defined by  $\mathcal{D}_{|\hat{u}\rangle}^M(i) = c^{-2} \cdot \mathcal{D}_{|\hat{u}\rangle}(s_i)$  where the normalization factor is  $c^2 = \sum_{i=0}^{p-1} \mathcal{D}_{|\hat{u}\rangle}(s_i)$ . The distribution  $\mathcal{D}_{|\hat{u}\rangle}^M$  can be thought of as induced by the experiment where  $|\hat{u}\rangle$  is measured, and if the value returned is not  $s_i$  for some  $i \in \{0, \dots, p-1\}$ ,

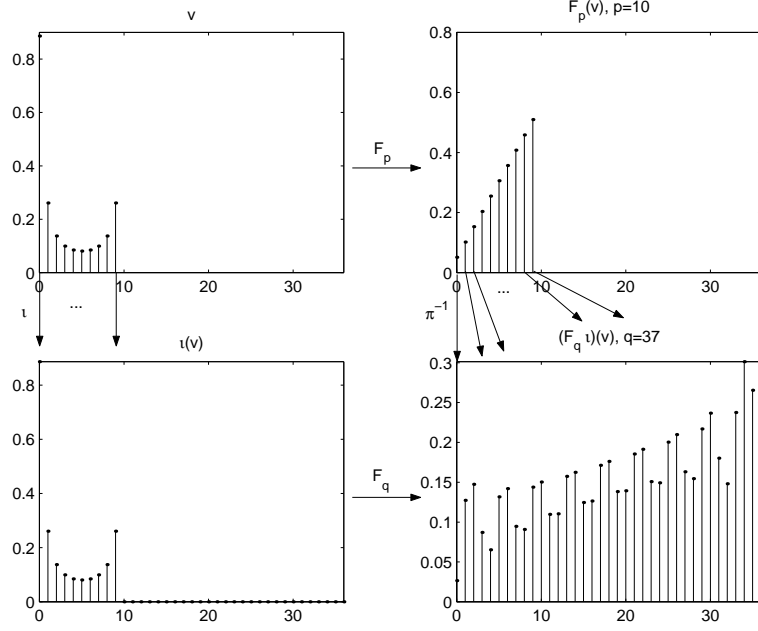


Figure 5.1: Graphs of magnitudes of amplitudes of four vectors.

throw it out and repeat. For this reason it is also important to have a lower bound on the probability of measuring some  $s_i$ .

**Corollary 5.1** *If  $q \geq (24 \cdot s \cdot \ln p) \cdot p$ , then  $|\mathcal{D}_{|\hat{u}\rangle}^M - \mathcal{D}_{|\hat{v}\rangle}|_1 \leq \frac{1}{s}$  and  $\Pr[\text{see some } s_i, i \in \{0, \dots, p-1\}] \geq \frac{p}{q} (1 - \frac{1}{s})$ .*

For the same reason as the lower bound, the probability of seeing some  $s_i$  is at most  $\frac{p}{q} (1 + \frac{1}{s})$ . Note that this implies that as  $q$  gets larger relative to  $p$ , the chance of actually measuring some  $s_i$  decreases, and that there is always at least a  $\frac{1}{24s \ln p}$  factor loss.

**Proof: (Corollary)** Let  $|\delta\rangle = \sqrt{\frac{q}{p}}P|\hat{u}\rangle - |\hat{v}\rangle$  and  $|\delta'\rangle = c \cdot P|\hat{u}\rangle - |\hat{v}\rangle$  (see Figure 5.2). The bound for  $|\delta'\rangle$  follows from the fact that  $\| |\delta'\rangle \| \leq \| |\delta\rangle \| + \| |\delta'\rangle - |\delta\rangle \| \leq 2\| |\delta\rangle \|$ , since  $|\delta'\rangle - |\delta\rangle$  cannot be longer than  $|\delta\rangle$ . By Theorem 5.1,  $\| |\delta\rangle \| \leq \frac{1}{8s}$ , so  $\| |\delta'\rangle \| \leq \frac{1}{4s}$ . The bound in the

corollary follows from the following lemma:

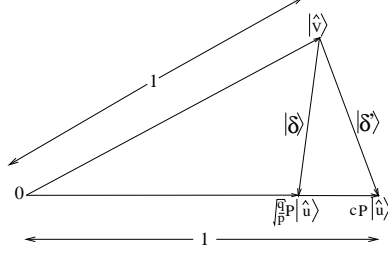


Figure 5.2: The notation used in the proof of Theorem 5.1.  $P|\hat{u}\rangle = \sum_{i=0}^{p-1} \hat{u}_{s_i}|i\rangle$ , and  $c \cdot \hat{u}$  is  $P|\hat{u}\rangle$  normalized to unit length.

**Lemma 5.1 ([8], Lemma 3.2.6)** *Let  $|\phi\rangle$  and  $|\psi\rangle$  be two unit length vectors with  $\| |\phi\rangle - |\psi\rangle \| \leq \epsilon$ . Then the total variation distance between the probability distributions resulting from measurements of  $|\phi\rangle$  and  $|\psi\rangle$  is at most  $4\epsilon$ .*

Since  $\Pr[\text{see some } s_i, i \in \{0, \dots, p-1\}] = \|P|\hat{u}\rangle\|^2$  and  $1 - \|\sqrt{\frac{q}{p}}P|\hat{u}\rangle\| \leq \|\sqrt{\frac{q}{p}}P|\hat{u}\rangle - |\hat{v}\rangle\| \leq \frac{1}{8s}$  by Theorem 5.1, we have  $\|P|\hat{u}\rangle\| \geq \sqrt{\frac{q}{p}}(1 - \frac{1}{8s})$ , and the bound follows.  $\blacksquare$

This method only extends to abelian groups with a constant number of factors. An arbitrary abelian group  $A$  is a direct product of cyclic groups  $\mathbb{Z}_{n_1} \times \dots \times \mathbb{Z}_{n_k}$ , and we could try a Fourier sampling approach using  $\mathcal{D}^M$  by using a larger cyclic group for each factor. That is, we would compute the Fourier transform  $F_{A'} = F_{m_1} \otimes \dots \otimes F_{m_k}$ , where  $m_j \geq n_j$  for all  $j$  and pick points  $s_i = (s_{i_1}^1, \dots, s_{i_k}^k)$ , where  $s^j$  rounds according to  $m_j$  and  $n_j$ . The problem is that we know that for each  $j$ , a factor of  $n_j/m_j$  is introduced, and this makes the probability of seeing a point  $s_i$  too small, unless the group only has a constant number of factors. For example, if  $n_j/m_j$  is only  $1/2$ , then the probability of seeing a point  $s_i$  drops below  $1/2^k$ .

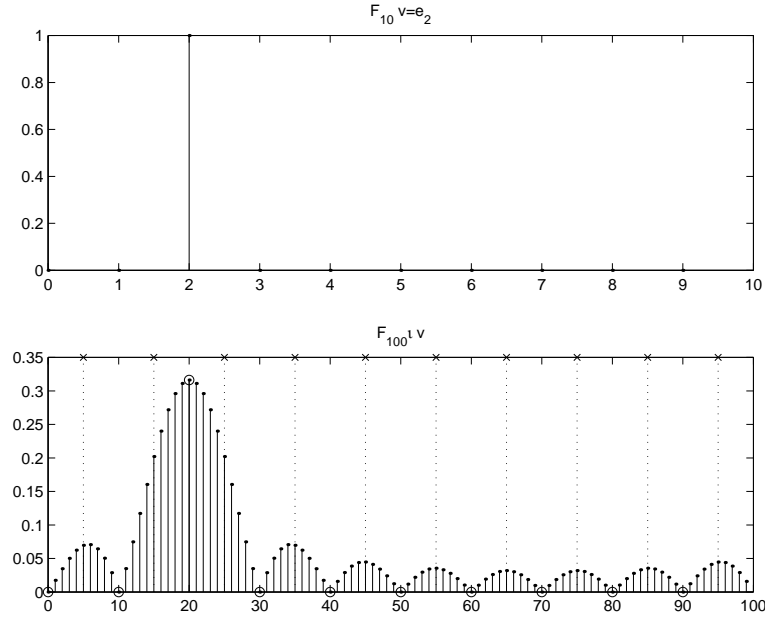


Figure 5.3:  $|\hat{v}\rangle = |2\rangle$  and  $q = 100$  in  $|\hat{u}\rangle$ . Buckets are marked off with dotted lines.

### 5.2.3 The Continuous Picture

In this section we will give the intuition behind the main theorem. We will show that the Fourier transform over  $\mathbb{Z}_p$  can be obtained as follows: regard the input vector  $|v\rangle$  as a vector over  $\mathbb{Z}$ , compute the inverse Fourier transform with respect to the circle group to get a continuous function, and then evaluate the function at  $p$  evenly spaced points to get the Fourier transform over  $\mathbb{Z}_p$ .

More formally, the Hilbert space  $l^2$  is the space of complex-valued sequences  $\{a_i | i \in \mathbb{Z}\}$  such that  $\sum_{i \in \mathbb{Z}} |a_i|^2 < \infty$ , with inner product  $\langle a, b \rangle = \sum_i a_i \bar{b}_i$ . The Hilbert space  $L^2$  is the space of complex-valued functions  $f$  on  $[0, 1)$  such that  $\int_0^1 |f(x)|^2 dx < \infty$ , with inner product  $\langle f, g \rangle = \int f(x) \overline{g(x)} dx$ . The inverse Fourier transform is an isometry (a vector space isomorphism preserving inner products) from  $l^2$  to  $L^2$ , defined by  $f(x) = \sum_{n=-\infty}^{\infty} a_n \omega^{nx}$ ,

where  $\{a_n\} \in l^2$ . See Conway [15] for more details. Terras [47] points out the connection between the continuous and discrete case, as we are discussing here.

Map a vector  $|v\rangle \in \mathbb{C}^p$  to the sequence  $\{u_i\} \in l^2$ , where  $u_i = v_i$  for  $0 \leq i < p$  and  $u_i = 0$  otherwise. The inverse Fourier transform of this sequence is  $f(x) = \sum_{i=0}^{p-1} v_i \omega^{ix}$ ,  $x \in [0, 1)$ . A coefficient  $\hat{v}_i$  of  $|\hat{v}\rangle$  is  $\hat{v}_i = \frac{1}{\sqrt{p}} f(\frac{i}{p})$  and similarly for  $|\hat{u}\rangle$ . If  $q$  is a multiple of  $p$ ,  $|\hat{u}\rangle$  will contain the exact same  $p$  amplitudes of  $|\hat{v}\rangle$  at multiples of  $\frac{q}{p}$ , up to a  $\sqrt{\frac{q}{p}}$  factor. When  $q$  is not a multiple of  $p$  our goal is similar: pick  $p$  amplitudes in  $|\hat{u}\rangle$  that approximate the amplitudes in  $|\hat{v}\rangle$ . In particular, we match  $\hat{v}_i$  with  $\hat{u}_{s_i}$ , for all  $i \in \{0, \dots, p-1\}$ . The reason this works can be seen in Figure 5.4. Ideally, we match  $\hat{v}_i$  with  $\hat{u}_{\frac{q}{p}i}$ , but we must round  $\frac{q}{p}i$  to the nearest integer. As  $q$  gets larger, the interval becomes more and more populated with evenly spaced points, so  $\sqrt{\frac{q}{p}} \hat{u}_{s_i} = \frac{1}{\sqrt{p}} f(\frac{i}{p} + \frac{\epsilon}{q})$  approaches  $\frac{1}{\sqrt{p}} f(\frac{i}{p})$  (in the Figure, scaling  $|\hat{u}\rangle$  to the width of  $|\hat{v}\rangle$ ,  $\frac{q}{p}s_i = i + \frac{q}{p}\epsilon$  approaches  $i$ ).

#### 5.2.4 Limitations of Zero-Filling

Our approach of Fourier sampling over a larger group and looking for points  $s_i$  may seem wasteful since we throw out much of the distribution, but unfortunately using all the points will not work. To see this, rather than using  $\mathcal{D}_{|\hat{u}\rangle}^M$  we will use the bucket distribution  $\mathcal{D}_{|\hat{u}\rangle}^B : \{0, \dots, p-1\} \rightarrow [0, 1]$ , defined by  $\mathcal{D}_{|\hat{u}\rangle}^B(i) = \sum_{t \in T} |\hat{u}_{s_i+t}|^2$ , where  $T = \{-\lfloor \frac{q}{2p} \rfloor + 1, \dots, \lfloor \frac{q}{2p} \rfloor\}$ . This uses all points in the distribution by counting any point in the interval  $\{s_i - \lfloor \frac{q}{2p} \rfloor + 1, \dots, s_i + \lfloor \frac{q}{2p} \rfloor\}$  as  $i$  (see Figure 5.3). Just by looking at Figure 5.4 it does not seem like this should work. The curve  $f$  is moving slowly between each integer point (it does for example not hit zero), so the bucket boundary probably represents some function of the two neighboring points. We now give a more formal argument.

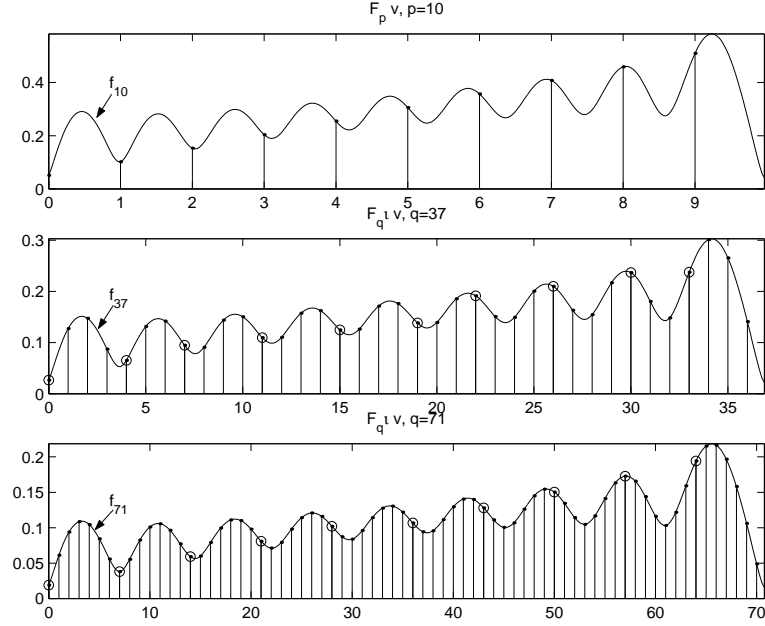


Figure 5.4: Scaled versions of  $f$  overlaid on  $|\hat{v}\rangle$  (from Figure 5.1), and on  $|\hat{u}\rangle$  for two different values of  $q$ .  $f_p(x) = \frac{1}{\sqrt{p}}f(px)$  and  $f_q(x) = \frac{1}{\sqrt{q}}f(qx)$ . Points  $s_i$  are marked with circles.

We will show that this method does not even work for a delta function. Given any  $q$  and any interval  $I = [a/q, b/q] \subseteq [0, 1]$  with  $a, b \in \mathbb{Z}$ , we have  $\Pr[\text{see } i \text{ s.t. } i/q \in I] = \frac{1}{q} \sum_{i=a}^b |f(\frac{i}{q})|^2$ . We can now let the point spacing get arbitrarily small while keeping the interval the same, and we have  $\lim_{n \rightarrow \infty} \frac{1}{nq} \sum_{i=na}^{nb} |f(\frac{i}{nq})|^2 = \int_{a/q}^{b/q} |f(x)|^2 dx$ . If  $f$  is nonzero outside of  $I$ , then the integral over  $I$  is less than one since  $f$  is continuous. If  $|\hat{v}\rangle = |0\rangle$ , then  $f(x) = \frac{1}{\sqrt{p}} \sum_{i=0}^{p-1} \omega^{ix}$ , and this is only zero for multiples of  $\frac{1}{p}$ . So for any  $q$ ,  $|\mathcal{D}_{|\hat{v}\rangle}(0) - \mathcal{D}_{|\hat{u}\rangle}^B(0)|_1 \geq c$ , for  $c$  a nonzero constant.

In the next section we will take an existing algorithm and show how to convert it into a Fourier sampling theorem where this problem does not occur.

5.2.5 Proof of Theorem 5.1

**Proof: (Theorem 5.1).** We will use [35] to improve the theorem statement in [31] and give a simpler proof.

Recall that we need to show that  $\| |\hat{v}\rangle - \sqrt{\frac{q}{p}} P |\hat{u}\rangle \| \leq \frac{\| |v\rangle \|}{8s}$ . Let  $|\delta\rangle = \sqrt{\frac{q}{p}} P |\hat{u}\rangle - |\hat{v}\rangle$  be the vector between the two vectors. We will start by computing a coefficient  $\delta_j$ . For a given  $j$ , we can use the linearity of the Fourier transform and write  $|\hat{u}\rangle = \hat{v}_j(F_q F_p^{-1}|j\rangle) + \sum_{c \neq j} \hat{v}_c(F_q F_p^{-1}|c\rangle)$ . Now we only have  $F_q F_p^{-1}$  of delta functions. Figure 5.5 and Lemma 5.2 describe this case.

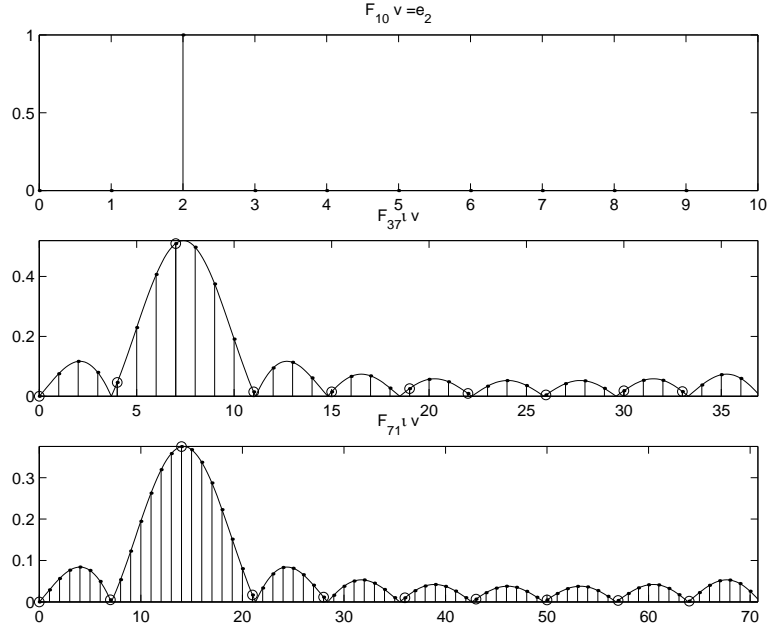


Figure 5.5:  $|\hat{v}\rangle = |2\rangle$ ,  $|\hat{u}\rangle = F_q F_p^{-1}|2\rangle$  for two values of  $q$ , the underlying continuous function, and points  $s_i$  marked with circles.

**Lemma 5.2** For a fixed  $j \in \{0, \dots, p-1\}$ , let  $|\hat{v}\rangle = |j\rangle$  and  $|\hat{u}\rangle = F_q F_p^{-1}|\hat{v}\rangle$ . Let  $\epsilon = s_j - \frac{q}{p}j$ .

Then:

- $\hat{u}_{s_j} = \sqrt{\frac{p}{q}} \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i\epsilon \frac{p}{q}}$  and  $\text{Re}(\hat{u}_{s_j}) \geq \sqrt{\frac{p}{q}} \left(1 - 5\frac{p^2}{q^2}\right)$
- For  $c \neq j$ ,  $\hat{u}_{s_c} = \sqrt{\frac{p}{q}} \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i(j-c+\frac{p}{q}\epsilon)}$  and  $|\hat{u}_{s_c}| \leq \sqrt{\frac{p}{q}} \frac{1}{|j-c|_p} \frac{p}{q}$

$$\text{where } |x|_p = \begin{cases} x \bmod p & \text{if } 0 \leq x \bmod p \leq p/2 \\ -x \bmod p & \text{otherwise} \end{cases}$$

The lemma states that when  $|\hat{v}\rangle$  is a delta function,  $|\hat{u}\rangle$  is relatively large at  $s_j$  and falls off as inverse distance at points  $s_c$ ,  $c \neq j$ , as shown in Figure 5.5.

So for a given  $j$ ,  $|\hat{u}\rangle = \hat{v}_j(F_q F_p^{-1}|j\rangle) + \sum_{c \neq j} \hat{v}_c(F_q F_p^{-1}|c\rangle)$ , and by Lemma 5.2 we have

$$\hat{u}_{s_j} = \hat{v}_j \sqrt{\frac{p}{q}} \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i\epsilon \frac{p}{q}} + \sum_{c \neq j} \sqrt{\frac{p}{q}} \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i(j-c+\frac{p}{q}\epsilon)}.$$

Now we bound  $\delta_j$ :

$$\begin{aligned} |\delta_j| &= \left| \sqrt{\frac{q}{p}} \hat{u}_{s_j} - \hat{v}_j \right| = \left| \hat{v}_j \left( \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i\epsilon \frac{p}{q}} - 1 \right) + \sum_{c \neq j} \hat{v}_c \left( \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i(j-c+\frac{p}{q}\epsilon)} \right) \right| \\ &\leq |\hat{v}_j| \left| \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i\epsilon \frac{p}{q}} - 1 \right| + \sum_{c \neq j} |\hat{v}_c| \left| \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i(j-c+\frac{p}{q}\epsilon)} \right| \leq |\hat{v}_j| 4\frac{p}{q} + \sum_{c \neq j} |\hat{v}_c| \frac{1}{|j-c|_p} \frac{p}{q} \end{aligned}$$

where if  $\alpha = \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i\epsilon \frac{p}{q}}$ , the last inequality follows from  $|1-\alpha|^2 = (1-\text{Re}(\alpha))^2 + \text{Im}(\alpha)^2 = 2(1-\text{Re}(\alpha))$  and Lemma 5.2.

We can now bound the length of  $|\delta\rangle$ :

$$\begin{aligned} \|\delta\rangle\|^2 &\leq \sum_j \left| |\hat{v}_j| 4\frac{p}{q} + \sum_{c \neq j} |\hat{v}_c| \frac{1}{|j-c|_p} \frac{p}{q} \right|^2 \leq \frac{\|\hat{v}\rangle\|^2}{p} \sum_j \left| 4\frac{p}{q} + \sum_{c \neq j} \frac{1}{|j-c|_p} \frac{p}{q} \right|^2 \\ &\leq \frac{\|\hat{v}\rangle\|^2}{p} \sum_j \left| 4\frac{p}{q} + \frac{p}{q} 2 \ln p \right|^2 \leq \frac{\|\hat{v}\rangle\|^2}{p} \left( 3\frac{p}{q} \ln p \right)^2 \sum_j 1 = \left( 3\frac{p}{q} \|\hat{v}\rangle\| \ln p \right)^2 \end{aligned}$$

The second inequality follows from the following observation. The left hand side can be viewed as the squared length of a matrix-vector product  $Mv$ , where the matrix and vector satisfy  $M_{jj} = 4\frac{p}{q}$ ,  $M_{jc} = \frac{1}{|j-c|_p}\frac{p}{q}$ , and  $v_c = |\hat{v}_c|$ . An upper bound is the operator norm of the matrix times the length of  $|\hat{v}\rangle$ . Since the matrix is symmetric, the norm is the magnitude of the largest eigenvalue. As the matrix is circulant, the eigenvalues are all of the form  $\sum_i \omega_p^{ij} M_{ij}$  for some  $j$ . Since the values of  $M$  are positive and real this is maximized when  $j = 0$ . Therefore the left hand side is maximized when  $|\hat{v}_j| = \frac{\|\hat{v}\|}{\sqrt{p}}$  for all  $j$ . ■

We end this section by proving Lemma 5.2.

**Proof:** (**Lemma 5.2**) The first bound is established as follows:

$$\hat{u}_{s_j} = \frac{1}{\sqrt{q}} \sum_{i=0}^{p-1} \frac{1}{\sqrt{p}} \omega_p^{-ij} \omega_q^{i(jq/p+\epsilon)} = \frac{1}{\sqrt{q}} \sum_{i=0}^{p-1} \frac{1}{\sqrt{p}} \omega_p^{-ij} \omega_p^{ij} \omega_q^{i\epsilon} = \sqrt{\frac{p}{q}} \frac{1}{p} \sum_{i=0}^{p-1} \omega_q^{i\epsilon}$$

Since  $|\epsilon| \leq \frac{1}{2}$ , for  $s_j$  we have  $\operatorname{Re}\left(\frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i\epsilon p/q}\right) \geq \cos(2\pi\epsilon p/q) \geq 1 - \frac{(2\pi\epsilon p/q)^2}{2} \geq 1 - 5(p/q)^2$ .

For the second bound we use the fact that the sum is a geometric series:

$$\hat{u}_{s_k} = \sqrt{\frac{p}{q}} \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i(k-j+\epsilon p/q)} = \sqrt{\frac{p}{q}} \frac{1}{p} \frac{1 - \omega_p^{\epsilon \frac{p}{q}}}{1 - \omega_p^{k-j+\epsilon \frac{p}{q}}}$$

In the numerator we have  $\left|1 - \omega_p^{\epsilon \frac{p}{q}}\right|^2 = \left|2\left(1 - \cos\left(2\pi\epsilon \frac{p}{q}\right)\right)\right| \leq \left(2\pi\epsilon \frac{p}{q}\right)^2 \leq \left(\pi \frac{p}{q}\right)^2$ . In the denominator we have:

$$\begin{aligned} \left|1 - \omega_p^{k-j+\epsilon \frac{p}{q}}\right| &= \sqrt{2\left(1 - \cos\left(2\pi\left(\frac{k-j}{p} + \frac{\epsilon}{q}\right)\right)\right)} = \left|2\sin\left(\pi\left(\frac{k-j}{p} + \frac{\epsilon}{q}\right)\right)\right| \\ &\geq 2\left|\sin\left(\pi\frac{k-j}{p}\right)\right| - \frac{4\pi\epsilon}{q} \geq 2\left|\sin\left(\pi\frac{k-j}{p}\right)\right| - \frac{2\pi}{q} \\ &\geq \frac{4}{p}|k-j|_p - \frac{2\pi}{q} \end{aligned}$$

The last inequality follows from the fact that  $\sin(x)$  is at least  $\frac{x}{\pi/2}$  when  $x \in [0, \pi/2]$  and is at least  $2 - \frac{x}{\pi/2}$  when  $x \in [\pi/2, \pi]$ .

So

$$\sqrt{\frac{p}{q}} \left| \frac{1}{p} \sum_{i=0}^{p-1} \omega_p^{i(k-j+\epsilon p/q)} \right| = \sqrt{\frac{p}{q}} \frac{1}{p} \frac{|1 - \omega_p^{\epsilon p/q}|}{|1 - \omega_p^{k-j+\epsilon p/q}|} \leq \sqrt{\frac{p}{q}} \frac{\pi p/q}{4|k-j|_p - \frac{\pi p}{q}} \leq \sqrt{\frac{p}{q}} \frac{1}{|k-j|_p} \frac{p}{q}$$

■

### 5.3 Repeating the Vector

In this section we cast results of Kitaev [36] and Cleve, Ekert, Macchiavello and Mosca [14] into our framework and derive another solution to Fourier sampling. Kitaev gives an algorithm for estimating the eigenvalue of a vector. In [14] they show that the quantum circuits for this algorithm and Shor's algorithm are the same. Using this we can take the power from Kitaev's algorithm and use it for our problem. In particular, from our point of view, the power essentially comes from taking the given superposition and repeating it. Given a superposition  $|v\rangle = \sum_{i=0}^{p-1} v_i |i\rangle$ , first compute  $|u\rangle = R_q |v\rangle$ . Then we have:

**Theorem 5.2** *If  $q = \Omega(p^3/\epsilon)$  then  $|\mathcal{D}_{|u\rangle}^B - \mathcal{D}_{|v\rangle}|_1 \leq \epsilon$ .*

This method immediately works for any abelian group by taking  $\epsilon$  small enough.

In the next section we will give a Fourier sampling theorem where  $q$  is smaller.

**Proof:** As shown in [36] and [14], given an eigenvector  $\sum_{i=0}^{q-1} \omega^{-\phi i} |i\rangle$  (for the map  $|i\rangle \rightarrow |i+1\rangle$ ), computing the Fourier transform results in a state that is highly concentrated near the basis state  $|\tilde{\phi}\rangle$ , where  $\tilde{\phi}$  is translated as the bits in the binary expansion of  $\phi$ . We have

**Lemma 5.3 ([39])**  $\Pr[|\frac{i}{q} - \phi| \leq \frac{k}{q}] \geq 1 - \frac{1}{2k-1}$ .

As usual, we are interested in understanding the delta function case  $|\hat{v}\rangle = |c\rangle$  and then using the linearity of the Fourier transform. So we must examine  $F_q R_q F_p^{-1} |c\rangle$ , where  $R_q$  repeats the state up to  $q$ . We now apply the theorem with  $\phi = c/p$  and  $k = \frac{q}{2p}$ , and we have  $\mathcal{D}_{|c\rangle}^{\text{B}}(c) \geq 1 - \frac{p}{q}$ , when  $q \geq 2p$ . We now use the linearity of the Fourier transform to show the same holds for any  $|\hat{v}\rangle$ . First examine the value of  $\mathcal{D}_{|\hat{v}\rangle}(c)$  for some  $c$ . It has probability at least  $1 - p/q$  minus the  $p/q$  that can interfere from each of the other  $p - 1$  delta functions. So there is at most  $p^2/q$  probability loss at each delta function, and we have  $\sum_i |\mathcal{D}_{|\hat{v}\rangle}^{\text{B}}(i) - \mathcal{D}_{|c\rangle}^{\text{B}}(i)| \leq p^3/q$ , and choosing  $q = \Omega(p^3/\epsilon)$  proves the theorem. ■

Repeating the given vector first in this way causes the superposition to be much more concentrated in the buckets. In fact, when  $q$  is an exact multiple of  $p$  it is easy to see the exact effect by looking at the underlying continuous function. In particular, if the original superposition is  $\sum_{i=0}^{p-1} v_i |i\rangle$ , then repeating first results in the state  $\frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} v_i |jp+i\rangle$ , and the Fourier transform has the associated continuous function

$$f(x) = \frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} \hat{v}_i \omega^{(jp+i)x} = \left( \frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \omega^{jpx} \right) \left( \sum_{i=0}^{p-1} \hat{v}_i \omega^{ix} \right).$$

Notice that the right factor is exactly the continuous function for the Fourier transform of the original state  $|v\rangle$ , and the effect of repeating can be factored out into the left scaling factor  $g(x) = (\frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \omega^{jpx})$ . Figure 5.6 shows how the functions  $g(x)$  works for different values of  $s$ . Note that if  $x$  is a multiple of  $1/p$ , then  $g(x) = \sqrt{s}$ , and if  $x$  is not a multiple of  $1/p$  but is a multiple of  $1/ps$ , then  $g(x) = 0$ .

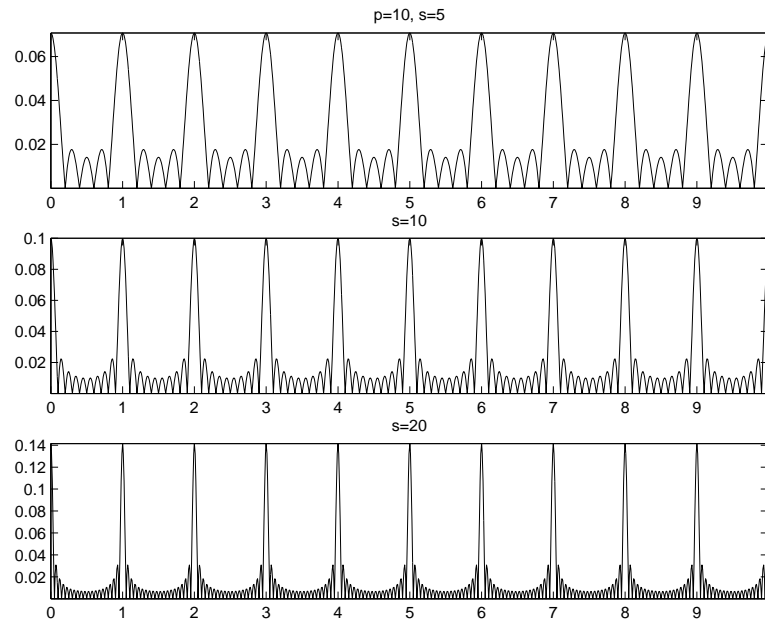


Figure 5.6: A graph of the function  $g(x) = \frac{1}{\sqrt{s}} \sum_{i=0}^{s-1} \omega^{ix}$ .

## 5.4 Repeating and Zero-Filling

### 5.4.1 The Main Theorem

In this section we will combine zero-filling and repeating the vector to get a better Fourier transform algorithm and a better Fourier sampling theorem. The main theorem of this section may be regarded as showing that the diagram

$$\begin{array}{ccc}
 \mathbb{C}^p & \xrightarrow{F_p} & \mathbb{C}^p \\
 \downarrow R_{ps} & & \downarrow S \\
 \mathbb{C}^{ps+r} & \xrightarrow{F_q} & \mathbb{C}^q
 \end{array}$$

“commutes approximately” when  $s$  is sufficiently large compared to  $p$ , and  $q = ps + r$  is sufficiently large compared to  $ps$ , where  $s$  is the number of copies of  $\mathbb{C}^p$ :  $\mathbb{C}^{ps} = \bigoplus_{i=0}^{s-1} \mathbb{C}^p$ ,

$R_{ps} = \sum_{i=0}^{s-1} \iota_i$  is the repeat map, where  $\iota_i$  is the inclusion map of  $\mathbb{C}^p$  into the  $i^{\text{th}}$  copy of  $\mathbb{C}^p$  in  $\mathbb{C}^{ps}$ , and  $S$  is a map similar to  $s$ , and will be described below. By “commutes approximately” we mean that for all  $v \in \mathbb{C}^p$ ,  $|SF_p|v\rangle - F_q R_{ps}|v\rangle| \leq \epsilon$ . We will then apply this to the quantum situation to achieve a new Fourier transform algorithm and a new Fourier sampling theorem.

Define  $T = \{-\lfloor q/2p \rfloor + 1, \dots, \lfloor q/2p \rfloor\}$  to index the consecutive basis vectors, and define the map  $S : \mathbb{C}^p \rightarrow \mathbb{C}^q$  by  $S|i\rangle = \sum_{t \in T} c_t |s_i + t\rangle$ . We will spell out the normalization constants  $c_t$  in the body of the proof – for now, we just point out that these constants are highly concentrated at the center of the interval.

Notice what this map does on an arbitrary vector: we have

$$\begin{aligned} S|\hat{v}\rangle &= \sum_{i=0}^{p-1} \hat{v}_i S|i\rangle = \sum_{i=0}^{p-1} \hat{v}_i \sum_{t \in T} c_t |s_i + t\rangle \\ &= \sum_{t \in T} c_t \left( \sum_{i=0}^{p-1} \hat{v}_i |s_i + t\rangle \right) \end{aligned}$$

So  $S|\hat{v}\rangle$  contains  $|T|$  copies of  $|\hat{v}\rangle$ , each weighted by some  $c_t$ . Furthermore, each copy is more or less evenly spread out in the interval  $\{0, \dots, q-1\}$ . We can now state the main theorem of this section.

**Theorem 5.3** *For any  $|v\rangle \in \mathbb{C}^p$ ,*

$$\|F_q R_{ps}|v\rangle - SF_p|v\rangle\| \leq \left( \frac{4ps}{q} + \frac{4 \ln p}{\sqrt{s}} \right) \| |v\rangle \|$$

We can now apply this to the quantum case to get an algorithm for the Fourier transform.

**Algorithm 5.1 (Approximate Fourier Transform)**

**Input:**  $|v\rangle = \sum_{i=0}^{p-1} v_i |i\rangle$

**Output:** An approximation of  $|\hat{v}\rangle = F_p |v\rangle$ .

1. Repeat the state  $s$  times:

$$|v\rangle \xrightarrow{F_s} \frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} |j\rangle \sum_{i=0}^{p-1} v_i |i\rangle \longrightarrow \frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} v_i |jp + i\rangle$$

2. Compute the Fourier transform over  $\mathbb{Z}_q$ :

$$\frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} v_i |jp + i\rangle \xrightarrow{F_q} \frac{1}{\sqrt{qs}} \sum_{d=0}^{q-1} \left( \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} v_i \omega_q^{(jp+i)d} \right) |d\rangle$$

3. Identify which copy of  $|\hat{v}\rangle$  each  $d$  is in:

$$|d\rangle \longrightarrow |c, t\rangle,$$

where  $d = \lfloor c \frac{q}{p} \rfloor + t$ . Call the resulting superposition  $\sum_{t \in T} \sqrt{p_t} |\hat{u}_t\rangle |t\rangle$ , where  $p_t$  is the probability of measuring  $t$ .

4. Measure the second register resulting in  $|\hat{u}_t\rangle |t\rangle$  with probability  $p_t$ .

**Corollary 5.2** Let  $|\hat{u}_t\rangle$  be as in Algorithm 5.1. For any  $\epsilon > 0$ , if  $s = \Omega\left(\frac{\log^2 p}{\epsilon^4}\right)$  and  $q = \Omega\left(\frac{ps}{\epsilon^2}\right)$ , then with probability  $1 - \epsilon^2$  over choices of  $t$ ,  $\| |\hat{v}\rangle - |\hat{u}_t\rangle \| \leq \epsilon$ .

**Proof:** We apply a standard averaging argument to conclude the bound. By Theorem 5.3 we have  $\sum_t \|\sqrt{p_t} |\hat{u}_t\rangle |t\rangle - c_t |\hat{v}\rangle |t\rangle\|^2 \leq \epsilon^4$ . Let  $B$  denote the set of bad  $t$ , where  $t \in B$  if  $\| |\hat{u}_t\rangle - c_t / \sqrt{p_t} |\hat{v}\rangle \|^2 > \epsilon^2$ . Suppose the probability of getting a bad  $t$  is at least  $\epsilon^2$ . Then we have  $\sum_{t \in B} \|\sqrt{p_t} |\hat{u}_t\rangle |t\rangle - c_t |\hat{v}\rangle |t\rangle\|^2 = \sum_{t \in B} p_t \| |\hat{u}_t\rangle - c_t / \sqrt{p_t} |\hat{v}\rangle \|^2 > \epsilon^2 \sum_{t \in B} p_t > \epsilon^4$ , contradicting Theorem 5.3.

Now assume we are in the good case when  $\|\widehat{u}_t\rangle - c_t/\sqrt{p_t}|\hat{v}\rangle\|^2 \leq \epsilon^2$ . Then by the triangle inequality we have  $\|\widehat{u}_t\rangle - |\hat{v}\rangle\| \leq \|\widehat{u}_t\rangle - c_t/\sqrt{p_t}|\hat{v}\rangle\| + \|c_t/\sqrt{p_t}|\hat{v}\rangle - |\hat{v}\rangle\| \leq \epsilon + |1 - c_t/\sqrt{p_t}| \leq 2\epsilon$ . The bound on  $1 - c_t/\sqrt{p_t}$  follows from the fact that  $c_t/\sqrt{p_t}|\hat{v}\rangle$  is at most  $\epsilon$  away from the unit vector  $|\widehat{u}_t\rangle$ . ■

**Corollary 5.3** *For any  $\epsilon$  and positive integer  $p$ , let  $n = \log p$ . Then Algorithm 5.1  $\epsilon$ -approximates the quantum Fourier transform over  $\mathbb{Z}_p$  and runs in time  $O(n \log n \log \log n + \log^2 \frac{1}{\epsilon})$ .*

The previous algorithm [36] for computing the Fourier transform for an arbitrary  $p$  takes time  $O(n^2)$ , so Algorithm 5.1 is strictly faster for polynomial approximations.

**Proof:** Coppersmith [16] gives an algorithm to  $\epsilon$ -approximate the Fourier transform over  $\mathbb{Z}_{2^n}$  in time  $O(n \log(\frac{n}{\epsilon}))$ , so choose  $s$  and  $q$  to be powers of two. Step 3 requires multiplying two  $n$  bit numbers which takes time  $n \log n \log \log n$ . ■

### 5.4.2 A Better Fourier Sampling Theorem

Theorem 5.3 can also be applied to achieve a better Fourier sampling theorem than in the last section.

#### Algorithm 5.2 (Fourier Sampling, $p$ known)

**Input:**  $|v\rangle = \sum_{i=0}^{p-1} v_i |i\rangle$

**Output:** A sample from a distribution that closely approximates  $\mathcal{D}_{|\hat{v}\rangle}$

1. Repeat the state  $s$  times:

$$|v\rangle \longrightarrow \frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} v_i |jp + i\rangle$$

2. Compute the Fourier transform over  $\mathbb{Z}_q$ :

$$\frac{1}{\sqrt{s}} \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} v_i |jp + i\rangle \xrightarrow{F_q} \frac{1}{\sqrt{qs}} \sum_{d=0}^{q-1} \left( \sum_{j=0}^{s-1} \sum_{i=0}^{p-1} v_i \omega_q^{(jp+i)d} \right) |d\rangle \stackrel{\text{def}}{=} |\hat{u}\rangle$$

3. Measure.

4. Round according to  $\mathcal{D}_{|\hat{u}\rangle}^B$ .

Recall from the last chapter that  $\mathcal{D}_{|\hat{v}\rangle}$  is the distribution induced by measuring  $|\hat{v}\rangle$ , and the bucket distribution  $\mathcal{D}_{|\hat{u}\rangle}^B : \{0, \dots, p-1\} \rightarrow [0, 1]$  is defined by  $\mathcal{D}_{|\hat{u}\rangle}^B(i) = \sum_{t \in T} |\hat{u}_{s_i+t}|^2$ , where  $T = \{-\lfloor \frac{q}{2p} \rfloor + 1, \dots, \lfloor \frac{q}{2p} \rfloor\}$ . This uses all points in the distribution by counting any point in the interval  $\{s_i - \lfloor \frac{q}{2p} \rfloor + 1, \dots, s_i + \lfloor \frac{q}{2p} \rfloor\}$  as  $i$  (see Figure 5.3).

**Corollary 5.4** *Let  $|\hat{u}\rangle$  be as in Algorithm 5.2. If  $s = \Omega\left(\frac{\log^2 p}{\epsilon^2}\right)$  and  $q = \Omega\left(\frac{ps}{\epsilon}\right)$  then*

$$|\mathcal{D}_{|\hat{v}\rangle} - \mathcal{D}_{|\hat{u}\rangle}^B|_1 \leq \epsilon.$$

**Proof:** Follows from Lemma 5.1 and Theorem 5.3. ■

In some of the most interesting quantum algorithms which use Fourier sampling neither  $p$  nor  $|v\rangle$  are known. However, it is often possible to generate a superposition of a chosen length  $q$  that consists of repetitions of  $|v\rangle$ , where  $q$  is not necessarily a multiple of  $p$ . That is, for an arbitrary integer  $q$  we assume we are given the superposition

$$c \sum_{i=0}^{l-1} v_{(i \bmod p)} |i\rangle,$$

where  $c$  is a normalization constant and  $\lceil l/p \rceil = s$ . Given this state as input, we will try to approximate  $\mathcal{D}_{|\hat{v}\rangle}$  as in Algorithm 5.2. This is not exactly possible though because we use the continued fractions algorithm, and that algorithm returns fractions in reduced terms.

We will now define the new distribution we will be close to.

Let  $p$  be the unknown period of  $|v\rangle$ . Define the reduced fractions distribution  $\mathcal{D}_{|\hat{v}\rangle}^{\text{RF}} : \{0, \dots, p-1\}^2 \rightarrow [0, 1]$  by  $\mathcal{D}_{|\hat{v}\rangle}^{\text{RF}}(c, r) = \mathcal{D}_{|\hat{v}\rangle}(cl)$  if  $rl = p$ . This distribution can be thought of as sampling some  $i$  from  $\mathcal{D}_{|\hat{v}\rangle}$  and making  $c/r$  the output, where  $c/r (= i/p)$  is  $i/p$  in reduced terms. For example, if  $p = 6$  and  $\mathcal{D}_{|\hat{v}\rangle}$  is the uniform distribution over  $\{0, 1, 2, 3, 4, 5\}$ , then  $\mathcal{D}_{|\hat{v}\rangle}^{\text{RF}}$  is a uniform distribution over  $\{0, (1, 6), (1, 3), (1, 2), (2, 3), (5, 6)\}$ . The distributions are basically the same, except some information is lost because the fractions are reduced. Of course, if  $p$  is prime, then nothing reduces and the distributions are just the same information-wise. The following algorithm approximates  $\mathcal{D}_{|\hat{v}\rangle}^{\text{RF}}$ .

**Algorithm 5.3 (Fourier Sampling,  $p$  unknown)**

**Input:**  $|v\rangle = \sum_{i=0}^{q-1} v_{(i \bmod p)} |i\rangle$ , an upper bound  $t$  on  $p$

**Output:** A sample from a distribution that closely approximates  $\mathcal{D}_{|\hat{v}\rangle}^{\text{RF}}$

1. Compute the Fourier transform over  $\mathbb{Z}_q$ :

$$\sum_{i=0}^{l-1} v_{(i \bmod p)} |i\rangle \xrightarrow{F_q} \frac{1}{\sqrt{q}} \sum_{d=0}^{q-1} \left( \sum_{i=0}^{l-1} v_{(i \bmod p)} \omega_q^{id} \right) |d\rangle \stackrel{\text{def}}{=} |\hat{u}\rangle$$

2. Measure, see  $d$ .

3. Use continued fractions on  $d/q$  and get  $c/r$ , which is  $i/p$  in reduced terms.

Given an upper bound  $t$  on  $p$ , let the continued fractions distribution  $\mathcal{D}_{|\hat{u}\rangle}^{\text{CF}} : \{0, \dots, t-1\}^2 \rightarrow [0, 1]$  be defined by  $\mathcal{D}_{|\hat{u}\rangle}^{\text{CF}}(c, r) = \Pr[c/r \text{ is output by Algorithm 5.3}]$ . The continued fractions algorithm used in Step 3 takes integers  $q$  and  $d$  with  $d < q$ , and a guarantee that  $|d - c\frac{q}{p}| \leq \frac{q}{2t^2}$ . In return it gives the unique fraction  $\frac{i}{p}$ , where  $p < t$ , in reduced terms in polynomial time.

**Corollary 5.5** *Let  $|\hat{u}\rangle$  be as in Algorithm 5.3 and let  $\lceil l/p \rceil = s$  be the number of repetitions.*

*If  $s = \Omega\left(\frac{t^2}{\epsilon^2 p}\right)$  and  $q = \Omega\left(\frac{ps}{\epsilon}\right)$ , then*

$$|\mathcal{D}_{|\hat{v}\rangle}^{RF} - \mathcal{D}_{|\hat{u}\rangle}^{CF}|_1 \leq \epsilon.$$

We may get a bogus value back from the continued fractions algorithm if the guarantee is not met, but notice that this corollary implies that the probability of this happening is less than  $\epsilon$ .

**Proof:** The notation here is defined at the beginning of the proof of Theorem 5.3. In addition let  $|\delta_0^{\text{IB}}\rangle$  be the inner bucket, i.e.  $|\delta_0^{\text{B}}\rangle$  restricted to integers between  $-q/2t^2$  and  $q/2t^2$ . We will show that  $\| |\delta_0^{\text{B}}\rangle - |\delta_0^{\text{IB}}\rangle \|^2 \leq \frac{t^2}{ps}$ , i.e. almost all the weight is close to the midpoint of the bucket. Then using Theorem 5.3 and the triangle inequality we have  $\| |\hat{u}\rangle - \sum_{i=0}^{p-1} \hat{v}_i |\delta_{0,i}^{\text{IB}}\rangle \| \leq \| |\hat{u}\rangle - \sum_{i=0}^{p-1} \hat{v}_i |\delta_{0,i}^{\text{B}}\rangle \| + \| \sum_{i=0}^{p-1} \hat{v}_i |\delta_{0,i}^{\text{B}}\rangle - \sum_{i=0}^{p-1} \hat{v}_i |\delta_{0,i}^{\text{IB}}\rangle \| \leq \frac{4ps}{q} + \frac{4 \ln p}{\sqrt{s}} + \frac{t}{\sqrt{ps}}$ . The corollary then follows from Lemma 5.1.

There is one technical point, which is that  $|\hat{u}\rangle$  is not exactly as stated in Theorem 5.3 because it is not necessarily an exact multiple of  $p$ . This does not change the bound, though. The distance between  $|\hat{u}\rangle$  and the same state repeated to the next multiple of  $p$  is at most  $1/\sqrt{s}$ , and adding a  $1/\sqrt{s}$  to the sum  $\frac{4ps}{q} + \frac{4 \ln p}{\sqrt{s}} + \frac{t}{\sqrt{ps}}$  does not change the bound.

We now bound  $\| |\delta_0^{\text{B}}\rangle - |\delta_0^{\text{IB}}\rangle \|$ . Let  $a = \lceil q/2t^2 \rceil$  and  $b = \lfloor q/2p \rfloor$ . By Equation 5.5 we have

$$\| |\delta_0^{\text{B}}\rangle - |\delta_0^{\text{IB}}\rangle \|^2 \leq 2 \sum_{i=a}^b |(\delta_0)_i|^2 \leq \sum_{i=a}^b \frac{q}{2ps} \frac{1}{i^2} \leq \frac{q}{2ps} \int_a^{b+1} \frac{1}{x^2} dx \leq \frac{q}{2ps} \frac{2t^2}{q} \leq \frac{t^2}{ps},$$

so  $\| |\delta_0^{\text{B}}\rangle - |\delta_0^{\text{IB}}\rangle \| \leq \frac{t}{\sqrt{ps}}$ . ■

Notice that this is easily applied to the factoring algorithm. A function is given with period  $p$  along with an upper bound  $t = N$ , the number trying to be factored. In this

case some order  $r$  of a periodic function must be found. Since this corresponds to  $|v\rangle = |c\rangle$  for some  $c \in \{0, \dots, r-1\}$ , the distribution  $\mathcal{D}_{|\hat{v}\rangle}$  will be uniform. Using Algorithm 5.3 allows us to sample from  $\mathcal{D}_{|\hat{u}\rangle}^{\text{CF}}$ , which is  $\epsilon$ -close to  $\mathcal{D}_{|\hat{v}\rangle}^{\text{RF}}$ . As soon as we measure some  $i$  relatively prime to  $r$ , we are done. We will give another application in Chapter 7.

### 5.4.3 Application: Many-to-One Periodic Functions

We will now give an application of the Fourier sampling theorem due to Hales [32]. We wish to generalize the period finding example by allowing a function to be many-to-one in its fundamental domain. That is, if the function has period  $p$ , it is not necessary that the function have distinct values in the domain  $\{0, \dots, p-1\}$ . By Corollary 5.5, we know that a starting point is to understand the distribution  $\mathcal{D}_{|\hat{v}\rangle}^{\text{RF}}$ , however as we will see in the algorithm below, there will be one additional complication.

We will first define a class of functions  $C_{1/d(n)}$  that contains a function  $f$  if one must change at least a  $1/d(n)$  fraction of its values to reduce its period. Such an  $f$  can be viewed as being  $1/d(n)$  robust with respect to its period. We will then show that for a given polynomial  $d(n)$  there is an efficient algorithm finding the period of any  $f \in C_{1/d(n)}$ .

More formally, let  $f : \mathbb{Z} \rightarrow \{0, \dots, 2^n - 1\}$  and  $g : \mathbb{Z} \rightarrow \{0, \dots, 2^n - 1\}$  be functions with periods  $p_f$  and  $p_g$ , respectively, each of length at most  $2^n$ . Define  $D(f, g)$  to be the fraction of points in  $\{0, \dots, p_f p_g - 1\}$  where  $f$  and  $g$  differ. Let  $C_{1/d(n)} = \{f | \forall g \text{ with } p_g < p_f, D(f, g) > 1/d(n)\}$ .

**Algorithm 5.4 (Many-to-One Period Finding)****Input:**  $f$ **Output:** *The period  $p$  of  $f$ .*

1. Repeat the following  $k = \text{poly}(\log p)$  times:

(a) Create the periodic state, measuring the second register:

$$\frac{1}{\sqrt{q}} \sum_{i=0}^{q-1} |i, f(i)\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{i: f(i)=a} |i\rangle |a\rangle$$

(b) Run Algorithm 5.3 to get a value  $c_i/r_i$ .

2. Output the least common multiple of the denominators  $r_1, \dots, r_k$ .

By the definition of  $f$  and  $g$  we have an upper bound on the period as required in Algorithm 5.3. Define the skewed distribution  $\mathcal{D}_f^S : \{0, \dots, p-1\}^2 \rightarrow [0, 1]$  by  $\mathcal{D}_f^S(c, r) = \Pr[c/r \text{ is output by Algorithm 5.4}]$ . This distribution is similar to  $\mathcal{D}_{|a\rangle}^{\text{CF}}$ , except that the repeated version of  $|v\rangle$  changes depending on the outcome of Step 1a.

**Theorem 5.4** *Let  $\epsilon = 1/d^3(n)$ ,  $s = \Omega(\frac{2^{2n}}{p\epsilon^2})$ ,  $q = \Omega(\frac{ps}{\epsilon})$ , and by  $k = 20d^2(n) \log n$ . Then Algorithm 5.4 outputs the period of  $f$  with probability at least  $3/4$ .*

**Proof:** The main problem is that we do not have a uniform superposition over  $\{0, \dots, p-1\}$ , so we may never get a value  $i/p$  with  $\gcd(i, p) = 1$  (i.e. we may never get a value  $r = p$ ). However, even though each denominator returned might never be  $p$ , the lcm of the denominators will be. The only way the lcm is missing some factor  $l$  of  $p$  is if every value returned is of the form  $cl/rl$ , where  $rl = p$ . The next lemma says that this does not happen with high probability.

**Lemma 5.4** ([32]) *Suppose that  $f \in C_{1/d(n)}$  has period  $p$ . Then for every prime  $l$  dividing  $p$ ,*

$$\Pr[l \text{ divides } j] < 1 - \frac{1}{8d^2(n)},$$

where the  $j$  are distributed according to  $\mathcal{D}_f^S$ .

By Corollary 5.3 and by choosing  $\epsilon = 1/d^3(n)$  we know that with probability at least  $1 - t\epsilon \geq 1 - 20/d(n)$  the condition for the continued fraction algorithm is met, so all of the fractions have a valid form (i.e.  $r_i$  divides  $p$ ). By Lemma 5.4,  $\Pr[l \text{ divides } j] \leq 1 - 1/8d^2(n) + \epsilon \leq 1 - 1/9d^2(n)$ . The probability that some prime divisor  $l$  of  $p$  occurs in every fraction  $i/p$  measured is at most  $n(1 - 1/9d^2(n))^k \leq 1/7$ , since there are at most  $n$  different possible primes dividing  $p$ . The lcm of the denominators therefore returns  $p$  with probability at least  $3/4$ . ■

The probability of success can be amplified because the answer can be tested. This restriction to functions in  $C_{1/d(n)}$ , for  $d(n)$  a polynomial is also necessary:

**Theorem 5.5** ([32]) *Let  $d(n) = o(2^n)$  be given. Suppose that  $A$  is a quantum algorithm which correctly computes the period of any  $f \in C_{1/d(n)}$  with probability at least  $3/4$ . Then  $A$  has worst-case run-time  $\Omega(\sqrt[4]{d(n)})$ .*

We finish this chapter by proving Theorem 5.3.

#### 5.4.4 Proof of Theorem 5.3

**Proof:** We wish to show that  $\|S|\hat{v}\rangle - |\hat{u}\rangle\| \leq \epsilon$ . Recall that  $|\hat{u}\rangle = F_q R_{ps}|v\rangle$ , so  $|\hat{u}\rangle = F_q R_{ps} F_p^{-1} |\hat{v}\rangle$ . We will use this fact to relate the two vectors. First we examine what

happens when  $|\hat{v}\rangle$  is some delta function  $|i\rangle$ , and then we combine the results using the linearity of the maps. We will define  $S$  in terms of the case when  $|\hat{v}\rangle = |0\rangle$ .

Let  $|\delta_i\rangle = F_q R_{ps} F_p^{-1} |i\rangle$ , let the bucket part of the vector be  $|\delta_i^B\rangle = \sum_{t \in T} \hat{u}_{s_i+t} |s_i + t\rangle$ , and let the tail part of the vector be  $|\delta_i^T\rangle = \sum_{t \notin T} \hat{u}_{s_i+t} |s_i + t\rangle$ , which implies that  $|\delta_i\rangle = |\delta_i^B\rangle + |\delta_i^T\rangle$ . Define  $|\delta_{0,i}^B\rangle$  to be  $|\delta_0^B\rangle$  shifted to  $s_i$ , and define  $S$  by  $S|i\rangle = |\delta_{0,i}^B\rangle$ . Writing the two vectors we are interested in in terms of these functions we have  $S|\hat{v}\rangle = \sum_{c=0}^{p-1} \hat{v}_c S|c\rangle = \sum_{c=0}^{p-1} \hat{v}_c |\delta_{0,c}^B\rangle$  and  $|\hat{u}\rangle = \sum_{c=0}^{p-1} \hat{v}_c F_q R_{ps} F_p^{-1} |c\rangle = \sum_{c=0}^{p-1} \hat{v}_c |\delta_c\rangle = \sum_{c=0}^{p-1} \hat{v}_c |\delta_c^B\rangle + \sum_{c=0}^{p-1} \hat{v}_c |\delta_c^T\rangle$ .

Using these we have:

$$\|S|\hat{v}\rangle - |\hat{u}\rangle\| = \left\| \sum_{c=0}^{p-1} \hat{v}_c (|\delta_{0,c}^B\rangle - |\delta_c^B\rangle) - \sum_{c=0}^{p-1} \hat{v}_c |\delta_c^T\rangle \right\| \quad (5.1)$$

$$\leq \left\| \sum_{c=0}^{p-1} \hat{v}_c (|\delta_{0,c}^B\rangle - |\delta_c^B\rangle) \right\| + \left\| \sum_{c=0}^{p-1} \hat{v}_c |\delta_c^T\rangle \right\| \quad (5.2)$$

$$\leq \frac{4ps}{q} \|v\| + \frac{4 \ln p}{\sqrt{s}} \|v\|. \quad (5.3)$$

$$(5.4)$$

The left summand of Equation 5.3 follows from Lemma 5.5 and the fact that the vectors in the set  $\{|\delta_c^B\rangle\}$  are pairwise orthogonal. The right summand of Equation 5.3 follows from Lemma 5.6.

The following lemma expresses the fact that excluding the tails, the vectors are close.

**Lemma 5.5**  $\|(|\delta_{0,c}^B\rangle - |\delta_c^B\rangle)\| \leq \frac{4ps}{q}$

**Proof:** We will give an upper bound for  $\|(|\delta_{0,c}\rangle - |\delta_c\rangle)\|$ , which is at least  $\|(|\delta_{0,c}^B\rangle - |\delta_c^B\rangle)\|$ .

For convenience we will compute the difference in the Fourier basis.

$$\begin{aligned}
\| |\delta_{0,c}\rangle - |\delta_c\rangle \|^2 &= \| F_q^{-1} |\delta_{0,c}\rangle - F_q^{-1} |\delta_c\rangle \|^2 = \frac{1}{ps} \sum_{i=0}^{ps-1} \left| (\omega_q^{-icq/p-i\epsilon} - \omega_p^{-ic}) \right|^2 \\
&= \frac{1}{ps} \sum_{i=0}^{ps-1} \left| \omega_p^{-ic} (\omega_q^{-i\epsilon} - 1) \right|^2 = \frac{1}{ps} \sum_{i=0}^{ps-1} |1 - \omega_q^{-i\epsilon}|^2 \leq |1 - \omega_q^{ps/2}|^2 \leq \left( \frac{4ps}{q} \right)^2
\end{aligned}$$

■

The following lemma expresses the fact that the tails are small and so will not interfere too much with the bound in the theorem.

**Lemma 5.6**  $\left\| \sum_{c=0}^{p-1} \hat{v}_c |\delta_c^T\rangle \right\| \leq \frac{4 \ln p}{\sqrt{s}} \|v\|$

**Proof:** Recall that

$$|\delta_c\rangle = F_q R_{ps} F_p^{-1} |c\rangle = \frac{1}{\sqrt{qps}} \sum_{d=0}^{q-1} \sum_{i=0}^{ps-1} \omega^{i(\frac{d}{q} - \frac{c}{p})} |d\rangle.$$

We have

$$|(\delta_j)_d| = \left| \frac{1}{\sqrt{qps}} \sum_{i=0}^{ps-1} \omega^{i(\frac{d}{q} - \frac{j}{p})} \right| \leq \frac{1}{\sqrt{qps}} \left| \frac{1 - \omega^{psd/q}}{1 - \omega^{(d-j\frac{q}{p})}} \right| \leq \sqrt{\frac{q}{ps}} \frac{2}{4|d - j\frac{q}{p}|_q}, \quad (5.5)$$

since  $|1 - \omega^{(d-j\frac{q}{p})}| = 2|\sin(\pi\frac{1}{q}(d-j\frac{q}{p}))| \geq \frac{4}{q}|d - j\frac{q}{p}|_q$ , which can be seen using the half angle formula and the fact that  $\sin(x) \geq 2x/\pi$ , when  $x \in [0, 1/2]$ .

Expanding the expression we have

$$\left\| \sum_{c=0}^{p-1} \hat{v}_c |\delta_c^T\rangle \right\|^2 = \sum_{d=0}^{q-1} \left| \sum_{\substack{j=0 \\ j \neq \lfloor dp/q \rfloor}}^{p-1} \hat{v}_j (|\delta_j^T\rangle)_d \right|^2 \leq \frac{q}{4ps} \sum_{d=0}^{q-1} \left| \sum_{\substack{j=0 \\ j \neq \lfloor dp/q \rfloor}}^{p-1} |\hat{v}_j| \frac{1}{|d - j\frac{q}{p}|_q} \right|^2$$

We apply Lemma 5.7 to pull  $\hat{v}_i$  out of the sum. The bound on the inner sum follows from the fact that  $|d - jq/p|_q \geq q/2p$ . We have

$$\sum_{\substack{j=0 \\ \neq \lfloor dp/q \rfloor}}^{p-1} \frac{1}{|d - j\frac{q}{p}|_q} \leq 2 \sum_{j=0}^{p-1} \frac{1}{j\frac{q}{p} + \frac{q}{2p}} \leq \frac{2p}{q} \sum_{j=0}^{p-1} \frac{1}{j + \frac{1}{2}} \leq \frac{2p}{q} \sum_{j=1}^p \frac{1}{j} \leq \frac{2p}{q} (\ln p + 1) \leq \frac{4p \ln p}{q}$$

Therefore,

$$\left\| \sum_{c=0}^{p-1} \hat{v}_c |\delta_c^T\rangle \right\|^2 \leq \frac{q}{p^2 s} \|v\|^2 \sum_{d=0}^{q-1} \left| \frac{4p \ln p}{q} \right|^2 \leq \frac{16 \ln^2 p}{s} \|v\|^2$$

■

**Lemma 5.7** For any vector  $x \in \mathbb{R}_{\geq 0}^p$ ,

$$\sum_{d=0}^{q-1} \left( \sum_{\substack{i=0 \\ i \neq \lfloor dp/q \rfloor}}^{p-1} \frac{x_i}{|d - \frac{q}{p}i|_q} \right)^2 \leq \frac{4\|x\|^2}{p} \sum_{d=0}^{q-1} \left( \sum_{\substack{i=0 \\ i \neq \lfloor dp/q \rfloor}}^{p-1} \frac{1}{|d - \frac{q}{p}i|_q} \right)^2$$

**Proof:** Let  $C$  be the matrix such that  $\|Cx\|^2 = \sum_{d=0}^{q-1} \left( \sum_{i=0, i \neq \lfloor dp/q \rfloor}^{p-1} \frac{x_i}{|d - \frac{q}{p}i|_q} \right)^2$ , i.e.

$C_{d,i} = \frac{1}{|d - \frac{q}{p}i|_q}$ . Our goal is to find the vector  $x_0$  that maximizes the sum and factor it out.

If  $C$  was circulant, we could use the usual argument (which we will use below) and conclude that the maximizing vector is parallel to the vector  $x$  where  $x_i = 1$  for all  $i$ . The matrix  $C$  is close to being circulant, however, and we will choose another matrix  $M$  and use it to achieve the bound.

We will compare  $\|Cx\|^2$  with the sum

$$\|Mx\|^2 \stackrel{\text{def}}{=} \sum_{t=0}^{\lceil q/p \rceil - 1} \|M_t x\|^2 = \sum_{t=0}^{\lceil q/p \rceil - 1} \sum_{k=0}^{p-1} \left( \sum_{\substack{i=0 \\ i \neq \lfloor dp/q \rfloor}}^{p-1} \frac{x_i}{|\frac{q}{p}k + t - \frac{q}{p}i|_q} \right)^2$$

First we will show that the vector parallel to the vector satisfying  $x_i = 1$  for all  $i$  maximizes the length of the vector  $Mx$ . Notice that for a fixed  $t$ ,  $M_t$  has entries  $M_{t,k,i} = \frac{1}{|t + \frac{q}{p}(k-i)|_q}$  and is circulant, i.e.,  $M_{t,k,i} = M_{t,k+1,i+1}$ . As the matrix is circulant, all its eigenvalues are of the form  $\sum_i \omega_p^{ij} M_{t,k,i}$  for some  $j$ . Since the values of  $M_t$  are positive and real this is maximized when  $j = 0$ . Therefore  $Mx$  is maximized when  $|x_j| = \frac{\|x\|}{\sqrt{p}}$  for all  $j$ .

Now observe that  $\frac{1}{2} \leq \frac{\|Cx\|^2}{\|Mx\|^2} \leq 2$ . This follows from the fact that for some sequence we have  $\sum_i \frac{1}{2l_i} \leq \sum_i \frac{1}{l_i+1}$ , which implies  $\sum_i \frac{1}{l_i} \leq 2 \sum_i \frac{1}{l_i+1}$ .

Let  $x_0$  maximize  $\|Mx\|^2$  over all vectors  $x$ . Then for any  $x$  we have  $\|Cx\|^2 \leq 2\|Mx\|^2 \leq 2\|Mx_0\|^2 \leq 4\|Cx_0\|^2$ . ■

■

## Chapter 6

# Fourier Sampling Coset States of Nonabelian Groups

In this chapter we analyze the HSP algorithm over nonabelian groups. The motivation is a reduction of graph isomorphism, a long standing open problem, to the HSP over the symmetric group  $S_n$ . In particular, we will analyze Fourier sampling coset states when the group is nonabelian. We show that Fourier sampling a polynomial number of coset states is enough to reconstruct hidden normal subgroups. This is just a generalization of the abelian case, since all subgroups of abelian groups are normal. There are two issues that we cannot address in general as we did in the abelian case. One is whether or not the quantum Fourier transform can be computed efficiently, and the other is whether or not we can efficiently compute the intersections of kernels of the group homomorphisms. These two questions must be answered on a group by group basis. We are basically analyzing what the Fourier transform can accomplish as a change of basis. We also show that Fourier

sampling a polynomial number of coset states is not enough to distinguish trivial from non-trivial subgroups of the symmetric group, a problem motivated by the graph isomorphism problem.

The algorithm for finding normal subgroups is as follows. Recall that  $M_i$  is a measurement of register  $i$ .

**Algorithm 6.1**

1. Repeat the following  $k = \text{poly}(\log(|G|))$  times:

a. Create a random coset state  $|cH\rangle$ :

$$|0\rangle \xrightarrow{F_G} \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle \xrightarrow{U_f} \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g, f(g)\rangle \xrightarrow{M_2} \frac{1}{\sqrt{|H|}} \sum_{h \in H} |ch\rangle$$

b. Fourier sample  $|cH\rangle$  and get a group homomorphism  $\rho$ . Call this distribution  $\mathcal{D}$ .

$$\frac{1}{\sqrt{|H|}} \sum_{h \in H} |ch\rangle \xrightarrow{F_G} \frac{1}{\sqrt{|H^\perp|}} \sum_{\rho \in H^\perp} \sum_{i,j=1}^{d_\rho} |\rho, i, j\rangle \xrightarrow{M_1} |\rho\rangle$$

2. Classically compute the intersection:  $H = \bigcap_{i=1}^k \ker \rho_i$ .

To define the Fourier transform over nonabelian groups we need some representation theory of groups. The rest of the chapter is organized as follows. In Section 6.1 some background on the representations of groups is given. In Section 6.2 a general formula is given for the distribution over representations when Fourier sampling coset states. In Section 6.3 it is shown that Fourier sampling coset states is enough to find normal hidden subgroups. In Section 6.4 it is shown that Fourier sampling a polynomial number of coset states is not enough to distinguish trivial from nontrivial subgroups in the symmetric group.

## 6.1 Representation Theory Background

The main tool used by polynomial time quantum algorithms is the Fourier transform. To define the Fourier transform (over a group) we require the basic elements of representation theory, defined below. See [34, 47] for more details.

**Representation.** A representation  $\rho$  of a group  $G$  is a homomorphism  $\rho : G \rightarrow GL(V)$ , where  $V$  is vector space over  $\mathbb{C}$ . Fixing a basis for  $V$ , each  $\rho(g)$  may be realized as a  $d \times d$  matrix over  $\mathbb{C}$ , where  $d$  is the dimension of  $V$ . As  $\rho$  is a homomorphism, for any  $g, h \in G$ ,  $\rho(gh) = \rho(g)\rho(h)$ . The *dimension*  $d_\rho$  of the representation  $\rho$  is  $d$ , the dimension of  $V$ .

A representation provides a means for investigating a group by homomorphically mapping it into a family of matrices. With this realization, the group operation is matrix multiplication and tools from linear algebra can be applied to study the group. We shall be concerned with complex-valued functions on a group  $G$ ; the representations of the group are relevant to this study, as they give rise to the Fourier transform for such functions.

**Irreducibility.** We say that a subspace  $W$  is an *invariant* subspace of a representation  $\rho$  if  $\rho(g)W \subseteq W$  for all  $g \in G$ . We assume, without loss of generality, that every  $\rho(g)$  is unitary, and, in particular, diagonalizable. Hence there are many subspaces fixed by an individual matrix  $\rho(g)$ . In order for  $W$  to be an invariant subspace for  $\rho$ , it must be simultaneously fixed under all  $\rho(g)$ .

The zero subspace and the subspace  $V$  are always invariant. If no nonzero proper

subspaces are invariant, the representation is said to be *irreducible*.

**Decomposition.** When a representation *does* have a nonzero proper invariant subspace  $V_1 \subset V$ , it is always possible to find a complementary subspace  $V_2$  (so that  $V = V_1 \oplus V_2$ ) which is also invariant. Since  $\rho(g)$  fixes  $V_1$ , we may let  $\rho_1(g)$  be the linear map on  $V_1$  given by  $\rho(g)$ . It is not hard to see that  $\rho_1 : G \rightarrow \text{GL}(V_1)$  is in fact a representation. Similarly, define  $\rho_2(g)$  to be  $\rho(g)$  restricted to  $V_2$ . Since  $V = V_1 \oplus V_2$ , the linear map  $\rho(g)$  is completely determined by  $\rho_1(g)$  and  $\rho_2(g)$ , and in this case we write  $\rho = \rho_1 \oplus \rho_2$ . In this case there is a basis for  $V$  so that each matrix  $\rho(g)$  is block diagonal with two blocks.

**Complete Reducibility.** Repeating the process described above, any representation  $\rho$  may be written as  $\rho = \rho_1 \oplus \rho_2 \oplus \cdots \oplus \rho_k$ , where each  $\rho_i$  is irreducible. In particular, there is a basis in which every matrix  $\rho(g)$  is block diagonal, the  $i$ th block corresponding to the  $i$ th representation in the decomposition.

**Characters.** The *character*  $\chi_\rho : G \rightarrow \mathbb{C}$  of a representation  $\rho$  is defined by  $\chi_\rho(g) = \text{Tr}(\rho(g))$ , the trace of the map. It is independent of the basis, and, as it turns out, completely determines the representation  $\rho$ .

**Orthogonality of Characters.** For two functions  $f_1$  and  $f_2$  on a group  $G$ , there is a natural inner product:  $\langle f_1, f_2 \rangle_G$  given by  $\frac{1}{|G|} \sum_g f_1(g) f_2(g)^*$ . The useful fact is the following: given the character  $\chi_\rho$  of any representation  $\rho$  and the character  $\chi_i$  of any irreducible representation  $\rho_i$ , the inner product  $\langle \chi_\rho, \chi_i \rangle$  is precisely the number of times the representation  $\rho_i$  appears in the decomposition of  $\rho$ . Since each  $\rho$  is unitary,

the inner product of two characters simplifies slightly:

$$\langle \chi_\rho, \chi_i \rangle_G = \frac{1}{|G|} \sum_{g \in G} \chi_\rho(g) \chi_i(g^{-1}).$$

**Restriction.** A representation  $\rho$  of a group  $G$  is also automatically a representation of any subgroup  $H$ . We refer to this *restricted* representation on  $H$  as  $\text{Res}_H \rho$ . Note that even representations which are irreducible over  $G$  may be reducible when restricted to  $H$ .

Up to isomorphism, a finite group has a finite number of irreducible representations. For a group  $G$ , we let  $\hat{G}$  denote this finite collection of irreducible representations. As mentioned above, any representation may be decomposed into a sum of representations in  $\hat{G}$ .

**Example 6.1** Fix a group  $G$  and a representation  $\rho$ . Let  $\rho_1, \dots, \rho_k$  be the irreducible representations of  $G$ . Desiring to know how  $\rho$  decomposes in these  $\rho_i$ , we compute

$$n_i = \langle \chi_\rho, \chi_i \rangle$$

for each  $i = 1, \dots, k$ . Then  $\rho = n_1 \rho_1 \oplus \dots \oplus n_k \rho_k$ , and after a unitary change of basis, the diagonal of the matrix  $\rho(g)$  consists of  $n_1$  copies of  $\rho_1(g)$ , followed by  $n_2$  copies of  $\rho_2(g)$ , and so on. ■

There are two representations that every group has:

**The Trivial Representation.** The trivial representation  $\mathbf{1}_G$  maps every group element  $g \in G$  to the 1 by 1 matrix (1). One feature of the trivial representation is that

$\sum_g \mathbf{1}_G(g)$  is the  $1 \times 1$  matrix ( $|G|$ ); this sum is the zero matrix for any other irreducible representation.

**The Regular Representation.** We take a vector space  $V$  with a basis vector  $e_g$  for every element  $g \in G$ . The regular representation  $\text{reg}_G : G \rightarrow \text{GL}(V)$  is defined by  $\text{reg}_G(g) : e_x \mapsto e_{gx}$  for any  $x \in G$ . It has dimension  $|G|$ . With the basis above, for any  $g \in G$ ,  $\text{reg}_G(g)$  is a permutation matrix.

An important fact about the regular representation is that it contains every irreducible representation of  $G$ . In particular, if  $\rho_1, \dots, \rho_k$  are the irreducible representations of  $G$  with dimensions  $d_1, \dots, d_k$ , then

$$\rho_{\text{reg}} = d_1 \rho_1 \oplus \dots \oplus d_k \rho_k,$$

that is, the regular representation contains each irreducible representation  $\rho_i$  exactly  $d_i$  times. Counting dimensions,

$$|G| = \sum_i d_i^2. \tag{6.1}$$

The main tool in quantum polynomial time algorithms is the Fourier transform.

**Definition 6.1** *Let  $f : G \rightarrow \mathbb{C}$ . The Fourier transform of  $f$  at the irreducible representation  $\rho$  is the  $d_\rho \times d_\rho$  matrix*

$$\hat{f}(\rho) = \sqrt{\frac{d_\rho}{|G|}} \sum_{g \in G} f(g) \rho(g).$$

We refer to the collection of matrices  $\langle \hat{f}(\rho) \rangle_{\rho \in \hat{G}}$  as the *Fourier transform* of  $f$ . Thus  $f$  is mapped into  $|\hat{G}|$  matrices of different dimensions. The total number of entries in these matrices is  $\sum d_\rho^2 = |G|$ , by equation 6.1 above. The Fourier transform is linear in  $f$ ;

with the constant used above (i.e.  $\sqrt{d_\rho/|G|}$ ) it is in fact unitary, taking the  $|G|$  complex numbers  $\langle f(g) \rangle_{g \in G}$  to  $|G|$  complex numbers organized into matrices.

A familiar case in computer science is when the group is cyclic of order  $n$ . Then the linear transformation (i.e. the Fourier transform) is a Vandermonde matrix with  $n$ -th roots of unity and the matrices are 1-by-1.

In the quantum setting we identify the superposition  $\sum_{g \in G} f_g |g\rangle$  with the function  $f : G \rightarrow \mathbb{C}$  defined by  $f(g) = f_g$ . In this notation,  $\sum_{g \in G} f(g) |g\rangle$  is mapped under the Fourier transform to  $\sum_{\rho \in \hat{G}, 1 \leq i, j \leq d_\rho} \hat{f}(\rho)_{i,j} |\rho, i, j\rangle$ . We remind the reader that  $\hat{f}(\rho)_{i,j}$  is a complex number. When the first portion of this triple is measured, we observe  $\rho \in \hat{G}$  with probability

$$\sum_{1 \leq i, j \leq d_\rho} |\hat{f}(\rho)_{i,j}|^2 = \|\hat{f}(\rho)\|^2$$

where  $\|A\|$  is the natural norm given by  $\|A\|^2 = \text{Tr} A^* A$ .

Let  $f$  be the indicator function of a left coset of  $H$  in  $G$ , i.e. for some  $c \in G$ ,

$$f(g) = \begin{cases} \frac{1}{\sqrt{|H|}} & \text{if } g \in cH \\ 0 & \text{otherwise.} \end{cases}$$

Our goal is to understand the Fourier transform of  $f$ . As mentioned above, the probability of observing  $\rho$  is  $\|\hat{f}(\rho)\|^2 = \sum_{i,j} |(\hat{f}(\rho))_{i,j}|^2$ . Our choice to measure only the representation  $\rho$  (and not the matrix indices) depends on the following key fact about the Fourier transform, also relevant to the abelian solution:

**Claim 6.1** *The probability of observing  $\rho$  is independent of the coset.*

**Proof:**  $\hat{f}(\rho) = \sqrt{\frac{d_\rho}{|G| |H|}} \sum_{h \in H} \rho(ch) = \rho(c) \sqrt{\frac{d_\rho}{|G| |H|}} \sum_{h \in H} \rho(h)$  and, since  $\rho(c)$  is a unitary matrix,

$$\|\hat{f}(\rho)\|^2 = \left\| \rho(c) \sqrt{\frac{d_\rho}{|G| |H|}} \sum_{h \in H} \rho(h) \right\|^2 = \left\| \sqrt{\frac{d_\rho}{|G| |H|}} \sum_{h \in H} \rho(h) \right\|^2.$$

■

Given this, we may assume that our function  $f$  is positive on the subgroup  $H$  itself, and zero elsewhere.

## 6.2 The Probability Distribution over Representations

The primary question is that of the probability of observing  $\rho$ . We have seen that this is determined by  $\sum_{h \in H} \rho(h)$  which, being a sum of the linear transformations  $\rho(h)$ , is a linear transformation. We begin by showing that it is a projection:

**Claim 6.2**  $\hat{f}(\rho)$  is a projection.

It is actually a scalar multiple of a projection, but we will ignore factors for now. With the right basis, then,  $\hat{f}(\rho)$  will be diagonal, and the diagonal entries will consist of ones and zeros. The probability of observing a particular representation  $\rho$  will then correspond to the number of ones appearing on the diagonal (i.e., on the dimension of the image of  $\hat{f}(\rho)$ ).

Given an irreducible representation  $\rho$  of  $G$ , we are interested in the sum of the matrices  $\rho(h)$  for all  $h \in H$ . Since we only evaluate  $\rho$  on  $H$ , we can instead consider  $\text{Res}_H \rho$  without changing anything. As mentioned before, though  $\rho$  is irreducible (over  $G$ ),  $\text{Res}_H \rho$  may not be irreducible on  $H$ . We may, however, decompose  $\text{Res}_H \rho$  into irreducible

representations over  $H$ . Then the Fourier transform of  $f$  at  $\rho$  is comprised of blocks, each corresponding to a representation in the decomposition of  $\text{Res}_H \rho$ . In particular, the matrix  $\sum_{h \in H} \rho(h)$  is:

$$U \begin{bmatrix} \sum_{h \in H} \sigma_1(h) & 0 & \cdots & 0 \\ 0 & \sum_{h \in H} \sigma_2(h) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{h \in H} \sigma_t(h) \end{bmatrix} U^* \quad (6.2)$$

for some unitary transformation  $U$  and some irreducible representations  $\sigma_i$  of  $H$  (with possible repetitions). We know that the sum is nonzero only when the irreducible representation is trivial, in which case it is  $|H|$ . Then the probability of observing  $\rho$  is

$$\begin{aligned} \|\hat{f}(\rho)\|^2 &= \left\| \sqrt{\frac{d_\rho}{|G|}} \frac{1}{\sqrt{|H|}} \sum_{h \in H} \rho(h) \right\|^2 = \frac{d_\rho}{|G|} \frac{1}{|H|} |H|^2 \langle \chi_\rho, \chi_{\mathbf{1}_H} \rangle_H \\ &= \frac{|H|}{|G|} d_\rho \langle \chi_\rho, \chi_{\mathbf{1}_H} \rangle_H. \end{aligned}$$

We have proved:

**Theorem 6.1** *For any subgroup  $H \leq G$ ,  $\|\hat{f}(\rho)\|^2 = \frac{|H|}{|G|} d_\rho \langle \chi_\rho, \chi_{\mathbf{1}_H} \rangle_H$ .*

Observe that one consequence of the theorem is that the probability of observing a representation  $\rho$  depends only on the character of  $\rho$ . It turns out that characters are class functions, i.e.  $\chi(g) = \chi(hgh^{-1})$  for any character  $\chi$  and  $h, g \in G$ . Hence conjugate subgroups ( $gHg^{-1}$  for some  $g \in G$  is a conjugate subgroup of  $H$ ) produce exactly the same distribution; this rules out using the paradigm of Algorithm 6.1 with measuring representations alone to solve the HSP for any group containing a subgroup that is not normal.

### 6.3 A Positive Result: Normal Subgroups

In this section we show that Fourier sampling a polynomial number of coset states suffices to reconstruct normal subgroups. Using Theorem 6.1, which describes the probability of measuring a representation, the main point is that when the subgroup is normal, the Fourier transform at each representation takes a nice form. Namely, it is basically either the identity or zero, depending on whether or not  $H$  is in the kernel<sup>1</sup> of  $\rho$ :

**Lemma 6.1** *Let  $H \trianglelefteq G$ . Then  $\hat{f}(\rho) = \begin{cases} c \cdot I & \text{if } H \subseteq \ker \rho, \text{ where } c = \frac{|H|}{|G|} d_\rho. \\ 0 \text{ matrix} & \text{otherwise} \end{cases}$*

**Proof:** The lemma follows from a simple application of Schur's lemma [47]. We will give the whole proof. Two alternative proofs are in [33]. If  $H \subseteq \ker \rho$  then the value of  $c$  follows from the discussion in the previous section.

Suppose  $H \not\subseteq \ker \rho$ . We will show that  $\hat{f}(\rho)$  must be the zero map. Let  $V = W_1 \oplus W_2$ , where  $H$  fixes  $W_1$  and kills  $W_2$ . Since  $\rho$  is irreducible over  $G$ , we can choose a vector  $w'_1 \in W_1$  and  $g \in G$  such that  $\rho_g w'_1 = w_1 + w_2$ , with  $w_i \in W_i$ , and  $w_2 \neq 0$ . Since  $H$  fixes  $W_1$  we have

$$w_1 + w_2 = \rho_g w'_1 = \rho_g \frac{1}{|H|} \sum_{h \in H} \rho_h w'_1 = \frac{1}{|H|} \sum_{h \in H} \rho_h \rho_g w'_1 = \frac{1}{|H|} \sum_{h \in H} \rho_h (w_1 + w_2) = w_1,$$

since  $H$  is normal in  $G$ . This is a contradiction unless  $\sum_{h \in H} \rho_h$  is the zero map. ■

We need one more fact to prove the theorem, which is that each time we sample we have a good chance of getting closer to  $H$ . Let  $H_i = \bigcap_{j=1}^i \ker \rho_j$ , the normal subgroup defined by the first  $i$  samples.

---

<sup>1</sup>We remind the reader that a representation  $\sigma$  is a homomorphism  $\sigma : G \rightarrow GL(V)$ . The *kernel* of  $\rho$  is the set  $\ker(\sigma) = \{g \in G | \sigma(g) = \mathbf{1}_V\}$  and is a normal subgroup of  $G$ , which we write  $\ker \sigma \trianglelefteq G$ .

**Claim 6.3** *If  $H_i \neq H$  then  $\Pr(H_{i+1} = H_i) \leq \frac{1}{2}$ ,*

**Proof:** By Lemma 6.1, Theorem 6.1 and Equation 6.1 we have:

$$\Pr(H_i \subseteq \ker \rho_{i+1}) = \sum_{\substack{\rho \in \hat{G} \\ H_i \subseteq \ker \rho}} \Pr(\rho) = \sum_{\substack{\rho \in \hat{G} \\ H_i \subseteq \ker \rho}} \frac{|H|}{|G|} d_\rho^2 = \frac{|H|}{|G|} \sum_{\rho \in \widehat{G/H_i}} d_\rho^2 = \frac{|H|}{|G|} \frac{|G|}{|H_i|} \leq \frac{1}{2}$$

where changing the sum follows from the fact that representations of  $G$  that map  $H_i$  to the identity can be identified with representations of  $G/H_i$ . ■

We now apply a Martingale Bound [40] to prove the theorem.

**Theorem 6.2** *Let  $\rho_1, \dots, \rho_k$  be independent random variables distributed according to  $\mathcal{D}$  with  $k = 4 \log_2 |G|$ . Then*

$$\Pr[H \neq \bigcap_i \ker \rho_i] \leq 2e^{-\log_2 |G|/8}.$$

**Proof:** For each  $i \in \{1, \dots, k\}$ , let  $X_i$  be the indicator random variable taking value 1 if  $H_i = H$  or  $H_{i+1} \neq H_i$ , and zero otherwise. The random variables  $X_1, \dots, X_k$  are not necessarily independent, but by the previous claim,  $\Pr[X_i = 0 | X_1, \dots, X_{i-1}] \leq \frac{1}{2}$ , and we can use a martingale bound. The function “sum” satisfies the Lipschitz condition, so we can apply Azuma’s Inequality to conclude that  $\sum_i X_i$  does not deviate much from its expected value, which is at least  $\frac{k}{2}$ . In particular, we have  $\Pr[|\sum_i X_i - \frac{k}{2}| \geq \lambda] \leq 2e^{-\lambda^2/2k}$ , so with  $\lambda = \log_2(|G|)$  we have  $\Pr\left[\sum_{i=0}^{k-1} X_i \leq \log_2(|G|)\right] \leq 2e^{-\log_2(|G|)/8}$ . ■

It is also easy to see how the algorithm works when  $H$  is not normal.

**Theorem 6.3** *For any subgroup  $H$  of  $G$ , Algorithm 6.1 returns the largest subgroup of  $H$  that is normal in  $G$ .*

**Proof:** This proof is due to Vazirani [50]. Let  $N$  be the intersection of the kernels so far. Let  $r_\rho$  be the rank of  $\hat{f}(\rho)$ , i.e.,  $r_\rho = \langle \chi_\rho, \chi_{tr} \rangle$ . When  $N \not\subseteq H$ , we will show that the probability of  $N$  being contained in the kernel of the next representation we measure is at most  $1/2$ , by showing that  $\sum_{\rho: N \subseteq \ker \rho} \frac{|H|}{|G|} d_\rho r_\rho \leq \frac{|N \cap H|}{|N|}$ , which is at most  $1/2$  when  $N \not\subseteq H$ . If the hidden subgroup had been  $HN$ , Theorem 6.1 would imply  $\sum_{\rho: \rho \in \hat{G}} \frac{|HN|}{|N|} d_\rho r'_\rho = 1$ , where  $r'_\rho$  is the number of times the trivial representation of  $HN$  appears in  $\rho$ . Note that  $r'_\rho = r_\rho$  when  $N \subseteq \ker \rho$ , since  $|H \cap N| \sum_{l \in HN} \rho(l) = (\sum_{h \in H} \rho(h)) (\sum_{n \in N} \rho(n))$ , and  $\rho(n)$  is the identity. Since  $|HN| \cdot |H \cap N| = |H| \cdot |N|$ , we have that  $\sum_{\rho: N \subseteq \ker \rho} \frac{|H|}{|G|} d_\rho r'_\rho \leq \sum_{\rho \in \hat{G}} \frac{|H|}{|G|} d_\rho r'_\rho \leq \frac{|H \cap N|}{|N|}$ , as desired.

Once  $N \subseteq H$ , the process stops. First, there is a unique largest subgroup  $H'$  of  $H$  that is normal in  $G$  since if there are two, their product is in  $H$  and is normal. Second, we measure representations  $\rho$  such that  $H' \subseteq \ker \rho$ . Let  $C$  be a set of coset representatives for  $H'$  in  $H$ . We have  $\sum_{h \in H} \rho(h) = (\sum_{c \in C} \rho(c)) (\sum_{h \in H'} \rho(h))$ , so by Lemma 6.1 we only see  $\rho$  if  $\sum_{h \in H'} \rho(h)$  is a multiple of the identity, i.e. only if  $H' \subseteq \ker \rho$ . ■

## 6.4 A Negative Result: Determining Triviality in the Symmetric Group

In this section we analyze the HSP algorithm when the group is the symmetric group  $S_n$ . We show that Fourier sampling a polynomial number of coset states is insufficient to distinguish the trivial subgroup from an order two subgroup. This implies that this algorithm cannot solve graph isomorphism.

**Example 6.2 Reducing graph isomorphism to the HSP over the symmetric group [37]**

Given two graphs  $G_1 = (V_1, E_1)$  and  $G_2 = (V_2, E_2)$  on vertex sets of size  $n/2$ , let  $G = G_1 \times G_2$  where  $V_1 = \{1, \dots, n/2\}$  and  $V_2 = \{n/2 + 1, \dots, n\}$ . Without loss of generality, assume the graphs are connected (otherwise looking at the complement of the graph suffices).  $G_1$  and  $G_2$  are isomorphic iff there exists a map  $\phi : V_1 \rightarrow V_2$  such that  $(i, j) \in E_1$  iff  $(\phi(i), \phi(j)) \in E_2$ .

Now define a function  $f : S_n \rightarrow T$  for a set  $T$  by  $f(\pi) = \pi G$ .  $f$  is an instance of the HSP. The hidden subgroup  $H$  is  $\text{Aut}(G)$ , the set of isomorphisms from  $G$  to  $G$ , called the automorphisms of the graph. We can write  $S_n$  as a sum of left cosets:  $S_n = \sum_i \sigma_i \text{Aut}(G)$ . Then for any  $\pi \in S_n$ , we have  $\pi = \sigma_i \phi$  for some  $i$  and  $\phi \in H$ .  $\pi_1$  and  $\pi_2$  are in the same coset iff they have the same set of edge violations. ■

We will now show that a special case of this problem cannot even be solved by Fourier sampling coset states. Consider the case when  $G_1$  and  $G_2$  are both rigid, that is,  $G_1$  and  $G_2$  have trivial automorphism groups. If  $G_1$  and  $G_2$  are not isomorphic,  $\text{Aut}(G) = \{e\}$ , and if they are isomorphic,  $\text{Aut}(G) = \{e, \tau\}$  where  $\tau$  swaps the vertices of  $G_1$  and  $G_2$  in some way, and so  $\tau$  is a product of  $n/2$  disjoint 2-cycles. Let  $\mathcal{D}_N$  and  $\mathcal{D}_I$  denote the two distributions induced by these two cases. It is enough to decide whether or not  $\text{Aut}(G)$  is trivial or nontrivial. We show that even for this particular case of graph isomorphism (and Graph Automorphism) the algorithm fails.

**Theorem 6.4**  $|\mathcal{D}_I - \mathcal{D}_N|_1 \leq 2^{-\Omega(n)}$ .

**Proof:** We present the proof from [28]. An alternative proof appears in [33]. When  $G_1 \not\approx G_2$ ,  $H = \{e\}$ , so  $\mathcal{D}_N(\rho) = \frac{d_\rho^2}{n!}$  by Theorem 6.1. When  $G_1 \approx G_2$ , and  $G_1$  and  $G_2$  are both connected and rigid,  $H = \{e, \tau\}$ . By Theorem 6.1

$$\mathcal{D}_I(\rho) = \frac{|H|}{|G|} d_\rho \langle \chi_1, \chi_\rho \rangle_H$$

$H$  has only two elements,  $e$  and  $\tau$ , hence

$$\langle \chi_1, \chi_\rho \rangle_H = \frac{1}{2}(\chi_\rho(e) + \chi_\rho(\tau)) = \frac{1}{2}(d_\rho + \chi_\rho(\tau)).$$

That is,  $\mathcal{D}_I(\rho) = \frac{d_\rho}{n!}(d_\rho + \chi_\rho(\tau))$  and so,

$$\sum_\rho |\mathcal{D}_I(\rho) - \mathcal{D}_N(\rho)| = \frac{1}{n!} \sum_\rho d_\rho |\chi_\rho(\tau)| \leq \frac{1}{n!} \sqrt{\sum_\rho d_\rho^2} \sqrt{\sum_\rho |\chi_\rho(\tau)|^2} \leq \frac{1}{\sqrt{n!}} \sqrt{\sum_\rho |\chi_\rho(\tau)|^2}$$

by the Cauchy-Schwartz Inequality and Equation 6.1.

To bound  $\sum_\rho |\chi_\rho(\tau)|^2$  we use the fact that the character table with the right normalization factors is a unitary matrix indexed by conjugacy classes and representations. In particular, the row for the conjugacy class of  $\tau$ ,  $\{\tau\}$ , has normalization factor  $\sqrt{|\{\tau\}|/|S_n|}$ .

The inner product of this row with itself is one, so we have:

$$\frac{1}{\sqrt{n!}} \sqrt{\sum_\rho |\chi_\rho(\tau)|^2} = \frac{1}{\sqrt{|\{\tau\}|}} = \sqrt{\frac{|C_{S_n}(\tau)|}{n!}} = \sqrt{\frac{2^{(n/2)}(n/2)!}{n!}} \leq 2^{-\Omega(n)},$$

where  $C_{S_n}(\tau) = \{\sigma \in S_n | \tau\sigma = \sigma\tau\}$  is the centralizer of  $\tau$ . ■

## Chapter 7

# Efficient Quantum Algorithms for Shifted Quadratic Character Problems

In this chapter we give efficient quantum algorithms for the Shifted Legendre Symbol Problem (SLSP) and its variants. There is some evidence that this is an intractable problem classically, and a closely related problem has been proposed as a cryptographic primitive [18]. In Section 7.1 we give an overview of the problems and related work. In Section 7.2 we give an efficient quantum algorithm for the SLSP and say how two variants follow as corollaries. In Section 7.3 we give an efficient quantum algorithm for the generalization of the SLSP to general fields, called the Shifted Quadratic Character Problem.

## 7.1 Definitions and Related Work

For a prime  $p$ , the *Legendre Symbol*  $\left(\frac{x}{p}\right)$  is defined to be 1 if  $x$  is a quadratic residue,  $-1$  if  $x$  is a quadratic nonresidue modulo  $p$ , and 0 if  $p|x$ . The Legendre Symbol can be extended in several ways. Here we will do so by defining it for rings  $\mathbb{Z}_N$  and finite fields  $\mathbb{F}_q$ . For an integer  $N = p_1 \cdots p_k$  the *Jacobi Symbol*  $\left(\frac{x}{N}\right)$  is defined by  $\left(\frac{x}{N}\right) = \left(\frac{x}{p_1}\right) \cdots \left(\frac{x}{p_k}\right)$ , where the  $\left(\frac{x}{p_i}\right)$  are Legendre Symbols and the product is over all the prime factors  $p_i$  of  $N$  (including repetitions). For a finite field  $\mathbb{F}_q$  and  $x \in \mathbb{F}_q$ , the *quadratic character*  $\chi(x)$  is 1 if  $x$  is a quadratic residue,  $-1$  if  $x$  is a quadratic nonresidue, and 0 if  $x = 0$ .

We can now define the problems solved in this chapter. The first problem is the basic example on which the others build.

**Definition 7.1 (Shifted Legendre Symbol Problem)** *Given an odd prime  $p$  and a function  $f_s : \mathbb{F}_p \rightarrow \{-1, 0, 1\}$  such that  $f_s(x) = \left(\frac{x+s}{p}\right)$  for some  $s \in \mathbb{F}_p$ , find  $s$ .*

The first variant extends the definition to rings.

**Definition 7.2 (Shifted Jacobi Symbol Problem)** *Given a squarefree odd integer  $N$  and a function  $f_s : \mathbb{Z}_N \rightarrow \{-1, 0, 1\}$  such that  $f_s(x) = \left(\frac{x+s}{N}\right)$  for some  $s \in \mathbb{Z}_N$ , find  $s$ .*

If the integer  $N$  is not square-free, the Shifted Jacobi Problem does not have a unique answer. For example, if  $N = p^2$  for a prime  $p$ , then  $\left(\frac{i}{N}\right) = \left(\frac{i}{p}\right)^2 = 1$  for all  $i$ . Instead we could define the task to find one of the values  $s'$  such that  $f_s(x) = \left(\frac{x+s'}{N}\right)$ . This problem is again efficiently solvable on a quantum computer.

The goal of the second variant is to also keep  $N$  unknown in the Shifted Jacobi Symbol Problem. Notice that the SLSP with  $p$  unknown is a special case of this problem.

**Definition 7.3 (Shifted Jacobi Symbol Problem, unknown  $N$ )** *Given an integer  $M$  and a function  $f_s : \mathbb{Z}_M \rightarrow \{-1, 0, 1\}$  such that  $f_s(x) = \left(\frac{x+s}{N}\right)$  for some integer  $N$ , with  $N^2 < M$ , find  $s$  and  $N$ .*

The last variant is a generalization to general finite fields.

**Definition 7.4 (Shifted Quadratic Character Problem)** *Given  $q = p^r$ , a power of an odd prime  $p$ , and a function  $f_s : \mathbb{F}_q \rightarrow \{-1, 0, 1\}$  such that  $f_s(x) = \chi(x+s)$  for some  $s \in \mathbb{F}_q$ , find  $s$ . Here  $\chi$  is the quadratic character of  $\mathbb{F}_q$ .*

Finding an efficient quantum algorithm for the Shifted Legendre Symbol Problem was originally posed as an open question by van Dam [48]. Damgard [18] has suggested using shifted Legendre and Jacobi sequences as pseudo-random bits. The seed to the pseudo-random number generator consists of  $s$  and  $p$ , and the output is the sequence  $\left(\frac{s}{p}\right), \left(\frac{s+1}{p}\right), \dots, \left(\frac{s+k}{p}\right)$ , where  $k$  is bounded by some polynomial in  $\log p$ . He shows that if Legendre sequences are unpredictable in a very weak sense, then Jacobi sequences (defined similarly) are unpredictable in a very strong sense. The SLSP with unknown  $p$  is at least as hard as the shifted Legendre sequence problem in the sense that solving the SLSP results in  $s$ , and then the next bits can be computed. However, it is potentially much easier to solve, because adaptive attacks are possible.

Many papers have studied the properties of Legendre sequences, as referenced in [18, 42]. Peralta [42] examines the distributions of the Legendre sequences. A corollary related to the problem of proving an oracle lower bound for the SLSP is: for a fixed plus/minus sequence of length  $t$ , the number of occurrences of that sequence in  $\left(\frac{1}{p}\right), \left(\frac{2}{p}\right), \dots, \left(\frac{p-1}{p}\right)$  is in the range of  $\frac{p}{2^t} \pm t(3 + \sqrt{p})$ . So from an information theoretic viewpoint, the best

bounds currently known do not rule out an algorithm that queries  $\log p$  consecutive values to reconstruct the offset.

## 7.2 An Algorithm for Prime Size Fields

In this section we give algorithms solving the Shifted Legendre Symbol Problem and variants when working over a finite field of prime size. The main ideas are contained in the algorithm for the Shifted Legendre Symbol Problem, and we can apply the same algorithm to solve the Shifted Jacobi Symbol Problem, and for the case when  $N$  is unknown.

The idea for the algorithm follows from a few known facts. Assume we start the algorithm in the standard way, i.e. by putting the function value in the phase to get  $|f_s\rangle = \sum_{i \in \mathbb{Z}_p} \left(\frac{i+s}{p}\right) |i\rangle$ . Assume the functions  $f_i$  are orthogonal (they are close to orthogonal). Define the matrix  $C$  where the  $i^{\text{th}}$  row is  $|f_i\rangle$ . Our quantum state  $|f_s\rangle$  is one of the rows, so  $C|f_s\rangle = |s\rangle$ . The issue now is how to efficiently implement  $C$ .  $C$  is a circulant matrix, i.e.  $c_{i,j} = c_{i+1,j+1}$ . The Fourier transform diagonalizes a circulant matrix:  $C = F_p(F_p^{-1}CF_p)F_p^{-1} = F_pDF_p^{-1}$ , where  $D$  is diagonal, so we can implement  $C$  if we can implement  $D$ . It turns out that the vector on the diagonal of  $D$  is the vector  $F_p|f_0\rangle$ , but  $|f_0\rangle$  is an eigenvector of the Fourier transform, so up to a global phase which we can ignore, we are done. To summarize: to implement  $C$ , we compute the Fourier transform, compute  $f_0$  into the phases (this is just the Legendre Symbol), and then compute the Fourier transform again (it is not important whether we use  $F_p$  or  $F_p^{-1}$ ). We will now present this algorithm step-by-step.

**Algorithm 7.1 (Shifted Legendre Symbol Problem)**

**Input:** An odd prime  $p$  and a function  $f_s$  with  $f_s(x) = \left(\frac{x+s}{p}\right)$ .

**Output:**  $s$ .

1. Compute the Fourier transform over  $\mathbb{Z}_p$  of  $|0\rangle$  and compute  $f_s$  into the phases, approximating:

$$\frac{1}{\sqrt{p-1}} \sum_{a \in \mathbb{F}_p} \left(\frac{a+s}{p}\right) |a\rangle$$

2. Compute the Fourier transform over  $\mathbb{Z}_p$ , yielding:

$$\frac{1}{\sqrt{p-1}} \sum_{b \in \mathbb{F}_p} \omega_p^{-bs} \left(\frac{b}{p}\right) |b\rangle$$

3. Compute  $f_0$  into the phases, approximating:

$$\frac{1}{\sqrt{p}} \sum_{b \in \mathbb{F}_p} \omega_p^{-bs} |b\rangle$$

4. Compute the Fourier transform over  $\mathbb{Z}_p$ ; this gives the answer  $|s\rangle$ .

**Theorem 7.1** *Algorithm 7.1 solves the Shifted Legendre Symbol Problem in two queries and polynomial time with probability exponentially close to one.*

**Proof:** The first step is a standard setup used in quantum algorithms. The only difference is that  $f_s$  evaluates to zero in one position. In this case, just treat it as a one. After this the state is exponentially close to the state shown. Recall that the Legendre Symbol  $\left(\frac{a}{p}\right)$  is zero when  $p|a$ , so one amplitude is zero.

The result of applying the Fourier transform is (where we replace  $a$  with  $a-s$ )

$$\frac{1}{\sqrt{p-1}} \sum_{a \in \mathbb{F}_p} \left(\frac{a+s}{p}\right) |a\rangle \longrightarrow \frac{1}{\sqrt{p-1}} \sum_{b \in \mathbb{F}_p} \frac{1}{\sqrt{p}} \sum_{a \in \mathbb{F}_p} \left(\frac{a}{p}\right) \omega_p^{b(a-s)} |b\rangle.$$

The sum  $b$  is over  $\mathbb{F}_p^*$  since the Legendre Symbol is 1 for half the nonzero elements and  $-1$  for the other half, so we can invert  $b$ . Factoring out the  $\omega_p^{-bs}$  term, using the change of variable  $c = ab$ , and using the facts that  $\left(\frac{cb^{-1}}{p}\right) = \left(\frac{c}{p}\right) \left(\frac{b^{-1}}{p}\right)$  and  $\left(\frac{b^{-1}}{p}\right) = \left(\frac{b}{p}\right)$  we have

$$\frac{1}{\sqrt{p-1}} \sum_{b \in \mathbb{F}_p} \left(\frac{b}{p}\right) \omega_p^{-bs} \left( \frac{1}{\sqrt{p}} \sum_{c \in \mathbb{F}_p} \left(\frac{c}{p}\right) \omega_p^c \right) |b\rangle$$

So we are left to evaluate  $\sum_{c \in \mathbb{F}_p} \left(\frac{c}{p}\right) \omega_p^c$ , which is the Gauss sum [13, 47], and is  $\sqrt{p}$  if  $p \equiv 1 \pmod{4}$ , and is  $\sqrt{-1}\sqrt{p}$  if  $p \equiv 3 \pmod{4}$ . Hence, up to a global constant which we can ignore, the state follows.  $\blacksquare$

**Corollary 7.1** *Algorithm 7.1 can be used to solve the Shifted Jacobi Symbol Problem.*

**Proof:** We start with the uniform superposition of  $\mathbb{Z}_N$  and calculate the function value  $f_s$  for each element:

$$\frac{1}{\sqrt{N}} \sum_{a \in \mathbb{Z}_N} |a, 0\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{a \in \mathbb{Z}_N} |a, \left(\frac{a+s}{N}\right)\rangle.$$

Next, we measure if the rightmost value is nonzero. If this is the case, which happens with probability  $\phi(N)/N$  (where  $\phi$  is Euler's phi-function obeying  $\phi(N) = |\mathbb{Z}_N^*|$ ), the state has collapsed to the superposition:

$$\frac{1}{\sqrt{\phi(N)}} \sum_{a \in \mathbb{Z}_N^*} |a, \left(\frac{a+s}{N}\right)\rangle.$$

Otherwise, we simply try the same procedure again. (The success probability  $\phi(N)/N$  has a lower bound of  $\Omega(1/\log(\log N))$ , see [3], hence we can expect to be successful after  $O(\log(\log N))$  trials.)

We continue with the reduced state by changing the phase of  $|a\rangle$  to  $\left(\frac{a+s}{N}\right)$  and

computing the function value again, giving

$$\frac{1}{\sqrt{\phi(N)}} \sum_{a \in \mathbb{Z}_N} \left( \frac{a+s}{N} \right) |a\rangle.$$

Let  $N = p_1 \cdot p_2 \cdots p_k$  be the prime decomposition of  $N$  such that  $\mathbb{Z}_N = \mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_k}$ .

Since  $\left(\frac{a}{N}\right) = \left(\frac{a}{p_1}\right) \cdot \left(\frac{a}{p_2}\right) \cdots \left(\frac{a}{p_k}\right)$ , we can just consider each  $p_j$  component separately (with  $s_1 \equiv s \pmod{p_1}$ ,  $s_2 \equiv s \pmod{p_2}$ , and so on). Hence, by performing the ‘inverse Chinese remainder’ map  $|a\rangle \rightarrow |a \bmod p_1, \dots, a \bmod p_k\rangle$ , we obtain the state

$$\sum_{a_1 \in \mathbb{Z}_{p_1}} \cdots \sum_{a_k \in \mathbb{Z}_{p_k}} \left( \frac{a_1 + s_1}{p_1} \right) \cdots \left( \frac{a_k + s_k}{p_k} \right) |a_1, \dots, a_k\rangle = \bigotimes_{j=1}^k \sum_{a_j \in \mathbb{Z}_{p_j}} \left( \frac{a_j + s_j}{p_j} \right) |a_j\rangle.$$

But now we use Algorithm 7.1 on each factor to get  $|s_1, \dots, s_k\rangle$ , after which the Chinese remainder theorem gives us the answer  $s$ . ■

We now give an algorithm for the case when  $N$  is unknown. In addition to using known techniques, the algorithm depends on the fact that sampling the Fourier transform of the shifted Legendre Symbol results in the uniform distribution.

**Algorithm 7.2 (Shifted Jacobi Symbol Problem, unknown  $N$ )**

**Input:** An integer  $M$  and a function  $f_s : \{0, \dots, M-1\} \rightarrow \{-1, 0, 1\}$  such that  $f_s(x) = \left(\frac{x+s}{N}\right)$  for some integer  $N$ , with  $N^2 < M$

**Output:**  $N$  and  $s$ .

1. Create the following state as in Corollary 7.1:

$$c \cdot \sum_{x=0}^{M-1} \left( \frac{x+s}{N} \right) |x\rangle$$

2. Compute the Fourier transform over  $\mathbb{Z}_M$ .

3. Measure, with outcome  $i$ , and use continued fractions on  $i$  and  $M$ , returning  $j/N$ .
4. Run Algorithm 7.1 using  $f_s$  and  $N$ .

**Theorem 7.2** *Algorithm 7.2 solves the Shifted Jacobi Symbol Problem with unknown  $N$  with high probability.*

**Proof:** Let  $|\psi_s\rangle = \frac{1}{\sqrt{\phi(N)}} \sum_{x=0}^{N-1} \left(\frac{x+s}{N}\right) |x\rangle$  be the state after the setup in Corollary 7.1, and let  $|\tilde{\psi}_s\rangle = c \sum_{x=0}^{M-1} \left(\frac{x+s}{N}\right)$  be the repeated version in Algorithm 7.2, where  $c$  is the normalizing constant. We can relate the distributions induced by Fourier sampling  $|\phi_s\rangle$  and  $|\tilde{\phi}_s\rangle$  using Corollary 5.5. If  $M = N$ , then Lemma 7.1 implies that  $i$  is uniformly distributed over  $\mathbb{Z}_N^*$ , and we would be done since the denominator returned by continued fractions is  $N$  in this case. However, this will still be the case even if  $M \neq N$ . If  $M$  is a multiple of  $N$  and if the Fourier transform of  $|\psi_s\rangle$  is  $\sum_{x=0}^{N-1} \alpha_x |x\rangle$ , then the Fourier transform of  $|\tilde{\psi}_s\rangle$  is  $\sum_{x=0}^{N-1} \alpha_x |M/N \cdot x\rangle$ , and  $N$  can be computed from samples. If  $M$  is any integer which is large enough compared to  $N$ , the distributions are  $\epsilon$ -close by Corollary 5.5. ■

**Lemma 7.1** *Let  $N$  be an odd squarefree integer. If we apply the quantum Fourier transform over  $\mathbb{Z}_N$  to the superposition of the states  $\left(\frac{x+s}{N}\right) |x\rangle$  for all  $x \in \mathbb{Z}_N$ , we have*

$$\frac{1}{\sqrt{\phi(N)}} \sum_{x \in \mathbb{Z}_N} \left(\frac{x+s}{N}\right) |x\rangle \xrightarrow{F_N} \frac{\sum_{z \in \mathbb{Z}_N} \left(\frac{z}{N}\right) \omega_N^z}{\sqrt{N \cdot \phi(N)}} \sum_{y \in \mathbb{Z}_N} \omega_N^{-sy} \left(\frac{y}{N}\right) |y\rangle.$$

**Proof:** First, we note that we can rewrite the output as

$$\frac{1}{\sqrt{N \cdot \phi(N)}} \sum_{y \in \mathbb{Z}_N} \sum_{x \in \mathbb{Z}_N} \left(\frac{x+s}{N}\right) \omega_N^{xy} |y\rangle = \frac{1}{\sqrt{N \cdot \phi(N)}} \sum_{y \in \mathbb{Z}_N} \omega_N^{-sy} \left[ \sum_{x \in \mathbb{Z}_N^*} \left(\frac{x}{N}\right) \omega_N^{xy} \right] |y\rangle,$$

by substituting  $x$  with  $x + s$  in the summation and using the fact that  $\left(\frac{x}{N}\right) = 0$  for all  $x \notin \mathbb{Z}_N^*$ .

The amplitudes between the square brackets depend on  $y$  in the following way. If  $y$  is coprime to  $N$ , then

$$\sum_{x \in \mathbb{Z}_N^*} \left( \frac{x}{N} \right) \omega_N^{xy} = \left( \frac{y^{-1}}{N} \right) \sum_{z \in \mathbb{Z}_N^*} \left( \frac{z}{N} \right) \omega_N^z,$$

where we used the substitution  $x = zy^{-1}$  and the multiplicativity of the Jacobi symbol.

Suppose  $N$  and  $y$  have a common, nontrivial, factor  $f$ . Let  $N = mf$  and  $y = rf$ . By the Chinese remainder theorem, there is a bijection between the elements  $x \in \mathbb{Z}_N$  and the coordinates  $(x \bmod m, x \bmod f) \in \mathbb{Z}_m \times \mathbb{Z}_f$ , which also establishes a one-to-one mapping between  $\mathbb{Z}_N^*$  and  $\mathbb{Z}_m^* \times \mathbb{Z}_f^*$ . This allows us to rewrite the expression as follows.

$$\begin{aligned} \sum_{x \in \mathbb{Z}_N^*} \left( \frac{x}{N} \right) \omega_N^{xy} &= \sum_{x \in \mathbb{Z}_{mf}^*} \left( \frac{x}{mf} \right) \omega_{mf}^{xrf} \\ &= \sum_{x_1 \in \mathbb{Z}_m^*} \left( \frac{x_1}{m} \right) \omega_m^{x_1 r} \sum_{x_2 \in \mathbb{Z}_f^*} \left( \frac{x_2}{f} \right). \end{aligned}$$

Because  $f$  is odd and squarefree  $\sum_{x \in \mathbb{Z}_f^*} \left( \frac{x}{f} \right) = 0$ , and hence the above term equals zero.

This concludes the proof of the lemma. ■

### 7.3 An Algorithm for General Finite Fields

Here will solve the general case of the Shifted Legendre Symbol Problem for a finite field  $\mathbb{F}_q$ . From now on  $q = p^r$ , with  $p$  an odd prime, and the degree  $r$  an integer. Field elements are polynomials and are computed modulo some irreducible polynomial of degree  $r$ . We will write  $a$  to mean  $\sum_{i=0}^{r-1} a_i x^i$ . The actual representation is just  $|a\rangle = |a_{r-1}, \dots, a_0\rangle$ .

**Lemma 7.2 (Trace-Fourier Transform over  $\mathbb{F}_q$ )** *The unitary mapping*

$$|a\rangle \longrightarrow \frac{1}{\sqrt{q}} \sum_{b \in \mathbb{F}_q} \omega_p^{\text{Tr}(ab)} |b\rangle$$

is computable in polynomial time.

**Proof:** First apply the map

$$|a\rangle \longrightarrow \bigotimes_{j=0}^{r-1} |\text{Tr}(ax^j)\rangle,$$

and then compute the Fourier transform over  $\mathbb{Z}_p^r$ . This gives us the final state

$$\bigotimes_{j=0}^{r-1} \frac{1}{\sqrt{p}} \sum_{b_j \in \mathbb{F}_p} \omega_p^{\text{Tr}(ax^j)b_j} |b_j\rangle = \frac{1}{\sqrt{q}} \sum_{b \in \mathbb{F}_q} \omega_p^{\text{Tr}(ab)} |b\rangle.$$

Now we will show how to compute the map above. First we will show that the map

$$|a\rangle \longrightarrow |\text{Tr}(a), \text{Tr}(ax), \dots, \text{Tr}(ax^{r-1})\rangle$$

is reversible. Let  $T(a) = [\text{Tr}(a), \text{Tr}(ax), \dots, \text{Tr}(ax^{r-1})]$ .  $T$  is a linear functional since  $\text{Tr}$  is, so if  $T(a) = T(b)$  then  $T(a - b)$  is the zero vector. We will show that  $T(a)$  is not the zero vector except for  $a = 0$ . Suppose  $T(a)$  is the zero vector. Since  $\text{Tr}$  is not the zero map, choose  $b \in \mathbb{F}_q$  such that  $\text{Tr}(b) \neq 0$ . Choose  $c_0, \dots, c_{r-1}$  such that  $\sum_i c_i ax^i = b$ . Then  $\text{Tr}(b) = \text{Tr}(\sum_i c_i ax^i) = \sum_i c_i \text{Tr}(ax^i) = 0$ , since  $\text{Tr}(ax^i) = 0$  for all  $i$ . But this is a contradiction by the choice of  $b$ . So  $T$  is 1-1.

We will now show that the map is computable in polynomial time. It is enough if  $a$  can be computed from  $\text{Tr}(a), \text{Tr}(ax), \dots, \text{Tr}(ax^{r-1})$ . But the equations  $\text{Tr}(a) = \sum_{j=0}^{r-1} a_j \text{Tr}(x^j)$ ,  $\text{Tr}(ax) = \sum_{j=0}^{r-1} a_j \text{Tr}(x^{j+1}), \dots, \text{Tr}(ax^{r-1}) = \sum_{j=0}^{r-1} a_j \text{Tr}(x^{j+r-1})$  are  $r$  linear equations in  $r$  unknowns, and the values  $\text{Tr}(a), \text{Tr}(ax), \dots, \text{Tr}(ax^{r-1})$  and  $\text{Tr}(x^j)$  for all  $j$  are known, so the coefficients  $a_j$  of  $a$  can be solved for using linear algebra. ■

Notice that the only properties of  $\text{Tr}$  used are that it is a linear functional, and that it could be computed efficiently. We can now give the algorithm for the general finite field case.

**Algorithm 7.3**

**Input:** A power of a prime  $q = p^r$  and a function  $f_s$  such that  $f_s(x) = \chi(x + s)$ .

**Output:**  $s$ .

1. Compute the Fourier transform over  $\mathbb{Z}_p^r$  of  $|0\rangle$  and compute the function value  $f_s$  into the phases, approximating:

$$\frac{1}{\sqrt{q-1}} \sum_{a \in \mathbb{F}_q} \chi(a + s) |a\rangle.$$

2. Compute the Trace-Fourier transform of Lemma 7.2 over  $\mathbb{F}_q$ , yielding:

$$\frac{1}{\sqrt{q-1}} \sum_{b \in \mathbb{F}_q} \chi(b) \omega_p^{\text{Tr}(-sb)} |b\rangle.$$

3. Uncompute the phases  $\chi(b)$  for  $b \neq 0$ , approximating:

$$\frac{1}{\sqrt{q}} \sum_{b \in \mathbb{F}_q} \omega_p^{\text{Tr}(-sb)} |b\rangle.$$

4. Compute the inverse Trace-Fourier transform over  $\mathbb{F}_q$ . This gives us the requested shift parameter as  $| - s \rangle$ .

**Theorem 7.3** *Algorithm 7.3 solves the Shifted Quadratic Character Problem over any finite field with two queries and in polynomial time with probability exponentially close to one.*

**Proof:** The proof is the same as the proof of Theorem 7.1. At step 2, we perform the Trace-Fourier transform to the state, yielding

$$\frac{1}{\sqrt{q-1}} \sum_{a \in \mathbb{F}_q} \chi(a + s) |a\rangle \longrightarrow \frac{1}{\sqrt{q-1}} \sum_{b \in \mathbb{F}_q} \frac{1}{\sqrt{q}} \sum_{a \in \mathbb{F}_q} \chi(a) \omega_p^{\text{Tr}(b(a-s))} |b\rangle.$$

The sum  $b$  is over  $\mathbb{F}_q^*$  since the quadratic character is 1 for half the nonzero elements and  $-1$  for the other half, so we can invert  $b$ . Factoring out the  $\omega_p^{\text{Tr}(-bs)}$  term, using the change of variable  $c = ab$ , and using the facts that  $\chi(cb^{-1}) = \chi(c)\chi(b^{-1})$  and  $\chi(b^{-1}) = \chi(b)$  we have

$$\frac{1}{\sqrt{q-1}} \sum_{b \in \mathbb{F}_q} \chi(b) \omega_p^{\text{Tr}(-bs)} \left( \frac{1}{\sqrt{q}} \sum_{c \in \mathbb{F}_q} \chi(c) \omega_p^{\text{Tr}(c)} \right) |b\rangle$$

So we are left to evaluate  $\sum_{c \in \mathbb{F}_q} \chi(c) \omega_p^{\text{Tr}(c)}$ , which is the Gauss sum [13, 47], and is  $\sqrt{p}$  if  $p \equiv 1 \pmod{4}$  and is  $\sqrt{-1}\sqrt{p}$  if  $p \equiv 3 \pmod{4}$ . Hence, up to a global constant which we can ignore, the state follows. ■

## Chapter 8

# Conclusion

We have given a new quantum Fourier sampling theorem and a new quantum Fourier transform algorithm for abelian groups. These results are based on the robustness of the Fourier transform with respect to changes in the underlying group – a property which might play a useful role in future quantum algorithms – for example, in the context of the Unique Shortest Vector Problem, where there is an underlying abelian group.

We have studied how the existing algorithm for the Hidden Subgroup Problem over abelian groups generalizes to the nonabelian case. This is motivated by the reduction of graph isomorphism to the HSP over the symmetric group. We characterize how the algorithm works and show that normal subgroups can be found. We have also shown that the algorithm cannot distinguish between a trivial and a nontrivial subgroup when the group is the symmetric group, so more must be done to solve graph isomorphism.

Finally, we solved the Shifted Legendre Symbol Problem, which appears to go beyond the framework of the HSP. The structure of the algorithm is different from that of

the HSP algorithm, and may provide a starting point for new quantum algorithms.

# Bibliography

- [1] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 284–293, El Paso, Texas, 4–6 May 1997.
- [2] Noga Alon and Joel H. Spencer. *The Probabilistic Method*. John Wiley & Sons, Inc., 1992.
- [3] Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory, Volume 1: Efficient Algorithms*. MIT Press, 1998.
- [4] Robert Beals. Quantum computation of Fourier transforms over symmetric groups. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 48–53, El Paso, Texas, 4–6 May 1997.
- [5] P. Benioff. Quantum mechanical hamiltonian models of turning machines. *J. Stat. Phys.*, 29:515–546, 1982.
- [6] C. H. Bennett. Logical reversibility of computation. *IBM J. of Research and Development*, 17:525–532, 1973.

- [7] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, October 1997.
- [8] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, October 1997.
- [9] A. Berthiaume and G. Brassard. The quantum challenge to structural complexity theory. In Osamu Barrington, David Mix; Brassard, Gilles; Hemachandra, Lane; Leivant, Daniel; Chair, Tim Long; Nisan, Noam; Royer, James; Watanabe, editor, *Proceedings of the 7th Annual Conference on Structure in Complexity Theory (SCTC '92)*, pages 132–137, Boston, MA, USA, June 1992. IEEE Computer Society Press.
- [10] A. Berthiaume and G. Brassard. Oracle quantum computing. *Journal of Modern Optics*, 41:2521–2535, 1994.
- [11] Dan Boneh and Richard J. Lipton. Quantum cryptanalysis of hidden linear functions (extended abstract). In Don Coppersmith, editor, *Advances in Cryptology—CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 424–437. Springer-Verlag, 27–31 August 1995.
- [12] B. H. Bransden and C. J. Joachain. *Introduction to Quantum Mechanics*. Longman Scientific & Technical, 1995.
- [13] Ronald J. Evans Bruce C. Berndt and Kenneth S. Williams. *Gauss and Jacobi Sums*. John Wiley & Sons, 1998.

- [14] R. Cleve, E. Ekert, C. Macchiavello, and M. Mosca. Quantum algorithms revisited. *Proc. Roy. Soc. Lond. A*, 454:339–354, 1998.
- [15] John B. Conway. *A Course in Functional Analysis*. Number 96 in Graduate Texts in Mathematics. Springer-Verlag, New York, NY, 2nd edition, 1990.
- [16] D. Coppersmith. An approximate fourier transform useful in quantum factoring. Technical Report RC19642, IBM, 1994.
- [17] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press and McGraw-Hill Book Company, 6th edition, 1992.
- [18] Ivan Bjerre Damgård. On the randomness of Legendre and Jacobi sequences. In *Advances in Cryptology—CRYPTO '88*, volume 403, pages 163–172. Springer-Verlag, 1990, 21–25 August 1988.
- [19] David Deutsch. Quantum theory, the church-turing principle and the universal quantum computer. *Proc. Roy. Soc. Lond. A*, 400:97–117, 1985.
- [20] David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proc. Roy. Soc. Lond. A*, 439:553–558, 1992.
- [21] Persi Diaconis and Daniel Rockmore. Efficient computation of the Fourier transform on finite groups. *J. Amer. Math. Soc.*, 3(2):297–332, 1990.
- [22] Mark Ettinger and Peter Høyer. A quantum observable for the graph isomorphism problem. Technical report, quant-ph/9901029, 1999.

- [23] Mark Ettinger and Peter Høyer. Quantum state detection via elimination. Technical report, quant-ph/9905099, 1999.
- [24] Mark Ettinger and Peter Høyer. On quantum algorithms for noncommutative hidden subgroups. *Advances in Applied Mathematics*, (25):239–251, 2000.
- [25] Mark Ettinger, Peter Høyer, and Emanuel Knill. Hidden subgroup states are almost orthogonal. Technical report, quant-ph/9901034, 1999.
- [26] R. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21:467–488, 1982.
- [27] Mikael Goldman and Alexander Russell. The computational complexity of solving systems of equations over finite groups. In *Fourteenth Annual IEEE Conference on Computational Complexity*, Atlanta, Georgia, 4–6 May 1999.
- [28] Michaelangelo Grigni, Leonard Schulman, and Umesh Vazirani. Quantum mechanical algorithms for the non-abelian hidden subgroup problem. Manuscript, 1997.
- [29] D.Y. Grigoriev. Testing the shift-equivalence of polynomials using quantum machines. *Theoretical Computer Science*, (180):217–228, 1997.
- [30] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 212–219, Philadelphia, Pennsylvania, 22–24 May 1996.
- [31] Lisa Hales and Sean Hallgren. Quantum fourier sampling simplified. In *Proceedings*

- of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 330–338, Atlanta, Georgia, 1–4 May 1999.
- [32] Lisa Hales and Sean Hallgren. An improved quantum fourier transform algorithm and applications. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, Redondo Beach, California, 12–14 November 2000.
- [33] Sean Hallgren, Alexander Russell, and Amnon Ta-Shma. Normal subgroup reconstruction and quantum computation using group representations. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 627–635, Portland, Oregon, 21–23 May 2000.
- [34] Joe Harris and William Fulton. *Representation Theory*. Number 129 in Graduate Texts in Mathematics. Springer-Verlag, New York, NY, 1991.
- [35] Peter Høyer. Simplified proof of the fourier sampling theorem. *Information Processing Letters*, 75:139–143, 2000.
- [36] Alexey Yu. Kitaev. Quantum measurements and the abelian stabilizer problem. Technical report, quant-ph/9511026, 1995.
- [37] Johannes Köbler, Uwe Schöning, and Jacobo Torán. *The Graph Isomorphism Problem: Its Structural Complexity*. Birkhäuser Boston Inc., Boston, MA, 1993.
- [38] G. L. Miller. Riemann’s hypothesis and tests for primality. *J. Comput. System Sci*, 13:300–317, 1976.
- [39] Michele Mosca and Artur Ekert. The hidden subgroup problem and eigenvalue estima-

- tion on a quantum computer. In *QCQS: NASA International Conference on Quantum Computing and Quantum Communications, QCQS*. LNCS, 1998.
- [40] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge, England, June 1995.
- [41] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
- [42] R. Peralta. On the distribution of quadratic residues and non-residues modulo a prime number. *Mathematics of Computation*, 58:433 – 440, 1992.
- [43] John Preskill. Lecture notes on quantum information and quantum computation. Technical report, Web address: [www.theory.caltech.edu/people/preskill/ph229](http://www.theory.caltech.edu/people/preskill/ph229).
- [44] Martin Rötteler and Thomas Beth. Polynomial-time solution to the hidden subgroup problem for a class of non-abelian groups. Technical report, quant-ph/9812070, 1998.
- [45] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.
- [46] Daniel R. Simon. On the power of quantum computation. *SIAM Journal on Computing*, 26(5):1474–1483, October 1997.
- [47] Audrey Terras. *Fourier Analysis on Finite Groups and Applications*. Number 43 in London Mathematical Society Student Texts. Cambridge University Press, 1999.

- [48] Wim van Dam. Quantum algorithms for weighing matrices and quadratic residues. Technical report, quant-ph archive no. 0008059, 2000.
- [49] Wim van Dam and Sean Hallgren. Efficient quantum algorithms for shifted quadratic character problems. Technical report, quant-ph archive no. 0011067, 2000.
- [50] U. V. Vazirani. Personal Communication, 2000.
- [51] Christof Zalka. On a particular non-abelian hidden subgroup problem.