Approved for the Division
(Mathematics)

__________________________
James Pommersheim
Acknowledgements

I must first thank my adviser Jamie Pommersheim: for introducing me to quantum computing and the Hidden Subgroup Problem, for guiding me through the dark forest of qubits in which I quickly found myself lost, and for being one of the truly luminous teachers during my time at Reed. I also give thanks to the other members of the Quantum Mechanics (Daniel Copeland, Ethan Edwards, Mikhail Lepilov, and Marcus Robinson), as well as all those Reed students who took part in Jamie’s topics class on quantum computing during the spring semester of 2012. Stephanie Bastek gets thanks for being a stellar roommate-proofreader. Finally, I thank Asif Shakeel for his thoughtful comments regarding chapter 3 and because his PhD thesis provided a launchpad for my research.
# Table of Contents

Introduction ................................................................. 1

Chapter 1: Quantum Computing ........................................... 3
1.1 Information storage ....................................................... 4
   1.1.1 State spaces ......................................................... 4
   1.1.2 Notation and composite systems ................................ 4
   1.1.3 State vectors ....................................................... 5
1.2 Information processing .................................................. 6
1.3 Information retrieval ..................................................... 8
1.4 The quantum Fourier transform ......................................... 9
1.5 Oracle problems ......................................................... 10

Chapter 2: The Hidden Subgroup Problem ................................. 13
2.1 History and overview ..................................................... 13
2.2 Statement of the problem ................................................. 14
2.3 The abelian HSP ......................................................... 14
2.4 The non-abelian HSP ..................................................... 16

Chapter 3: The HSP for $\mathbb{Z}_N \rtimes \mathbb{Z}_p$ ................................ 19
3.1 The metacyclic group $\mathbb{Z}_N \rtimes \mathbb{Z}_p$ ......................... 19
3.2 Overview of the algorithm ............................................. 20
3.3 Preliminaries ............................................................ 23
3.4 Single-query algorithm .................................................. 25
3.5 Multi-query algorithm .................................................... 28
3.6 Analysis of probability of success ..................................... 30
   3.6.1 A lemma on the distribution of $\vec{d}$’s .......................... 31
   3.6.2 $N$ prime ......................................................... 32
   3.6.3 General $N$ ....................................................... 32
3.7 Conclusion ............................................................... 33

Chapter 4: Representation Theory and the HSP for $\mathbb{Z}_N \rtimes \mathbb{Z}_p$ .............................. 35
4.1 The Fourier transform over a group .................................. 35
4.2 Irreducible representations of $\mathbb{Z}_N \rtimes_k \mathbb{Z}_p$ ................ 37
4.3 The Fourier transform over $\mathbb{Z}_N \rtimes_k \mathbb{Z}_p$ ..................... 39
Abstract

In this thesis we explore the Hidden Subgroup Problem in quantum computing. While there exist efficient quantum algorithms that solve the Hidden Subgroup Problem for many families of groups, there is no known general polynomial-time solution. Our main new result, proved in chapter 3, shows that if a certain proposed algorithm for the dihedral Hidden Subgroup Problem can be efficiently implemented, then in fact there exists an efficient quantum solution to the Hidden Subgroup Problem for arbitrary semidirect products of a cyclic group by a cyclic group of prime order.
Introduction

In 1994, Peter Shor [22] shocked the computer science community by demonstrating that a quantum computer, if it were ever built, could quickly factorize integers. Practically, Shor’s algorithm threatened to break the most widely used public-key cryptography systems and immediately spurred government and industry to invest in quantum computing research. Philosophically, Shor’s algorithm represented a potential counterexample to the Extended Church-Turing Thesis.\(^1\) Mathematically, Shor’s algorithm, which employs variously number theory, linear algebra, probability and analysis, laid bare the richness of the theory of quantum computing and prompted mathematicians to explore its capabilities and limitations. Almost twenty years later, there have been few breakthroughs in quantum computing comparable to Shor’s algorithm. Indeed, even as engineers are testing small-scale quantum computers that can factor 15 or 21, there exist only a handful of significant quantum algorithms. As Amit Hagar [10] writes in the Stanford Encyclopedia of Philosophy, “quantum computing is a domain where experimentalists find themselves ahead of their fellow theorists.”

This thesis represents one theorist fighting back. In the following pages, I attempt to develop an algorithm for solving a particular case of the so-called Hidden Subgroup Problem (HSP), a natural algebraic generalization of Shor’s problem of factorizing integers. The HSP is a highly active area of research in quantum computing: it has implications for certain proposed cryptographic systems and is of general computer scientific interest because of its relation to classically intractable problems like the graph isomorphism problem. While my research has not led to a new, unconditionally efficient solution to the HSP, it does expose an interesting connection between the dihedral HSP and the HSP for more general semidirect products of cyclic groups.

Throughout the thesis, we assume familiarity with basic results from linear algebra and group theory. Representation theory is used only in chapter 4; Appendix A quickly explains all the representation-theoretic terms and theorems we need.

\(^1\)The Extended Church-Turing Thesis is the claim that all reasonable physical models of a computer are polynomial-time equivalent to a Turing machine; see [10], §1.1.
Chapter 1

Quantum Computing

In this chapter we describe the standard model of quantum computing and present its postulates, with the aim of developing a quantum algorithm that solves some classically hard mathematical problem. These postulates are justified by well-documented physical evidence but we will not focus on providing any such justification here. For a comprehensive treatment of the theory of quantum computing, with perspectives from physics, mathematics, computer science and information theory, we direct the reader’s attention to Nielsen and Chaung [15].

Before formally describing the model, it will help to give a brief overview of quantum computers to understand why they are potentially interesting. A quantum computer stores information as quantum bits, or qubits, where a qubit is mathematically notated as a vector \(|\phi\rangle = \alpha |0\rangle + \beta |1\rangle\). Unlike classical bits, qubits can take values “in-between” 0 and 1. Here \(\alpha\) and \(\beta\) are complex coefficients which we call amplitudes, and so a single qubit could be said to store an infinite amount of information. However, this last assertion is extremely misleading because we are not free to arbitrarily access the amplitudes of our qubits. Rather, when we measure a qubit, we obtain only classical information (either 0 or 1). The real values \(|\alpha|^2\) and \(|\beta|^2\) determine the probabilities of measuring 0 or 1 respectively.

The naïve account of quantum computing contends that a quantum algorithm can solve an intractable problem because “it tests all the possibilities at once.” But this description is misleading. If we used a single system of qubits to compute all possible solutions to some problem and then attempted to read off the solution by measuring this system, instead we would only measure the result of a random computation. This is no better than the classical probabilistic approach of randomly guessing the answer and checking if it works. However, quantum computers have one tremendous advantage over classical probabilistic computers: unlike classical probabilities, which are nonnegative reals, the amplitudes of a quantum state are complex numbers and thus can cancel out. The true explanation for why quantum algorithms work is that they exploit the structure of the problems they solve and cleverly manipulate their quantum states so that the amplitudes of bad solutions destructively interfere while the amplitudes of the correct solutions constructively interfere. With this perhaps fuzzy intuition about the potential of quantum computing in mind, we now systemically describe the way a quantum computer stores, processes, and retrieves information.
1. Information storage

1.1 State spaces

Quantum computers store information in quantum systems; indeed, it is through quantum mechanical phenomena such as superpositions and entanglement that they are able to achieve any computational advantage over classical computers. Von Neumann, in rigorously establishing a mathematical framework for quantum mechanics, realized that the proper setting for a quantum system is a Hilbert space. Throughout this thesis, we will consider our Hilbert spaces to be finite-dimensional complex inner-product spaces.

Postulate 1. (State spaces)
The state space of a quantum system is a finite-dimensional complex Hilbert space \( \mathcal{H} \). That is, \( \mathcal{H} \) is a finite-dimensional vector space over \( \mathbb{C} \) equipped with an inner product \( \langle \ , \rangle : \mathcal{H} \times \mathcal{H} \rightarrow \mathbb{C} \) such that for all \( x,y,z \in \mathcal{H} \) and \( a,b \in \mathbb{C} \),

1. \( \langle ax + bz, y \rangle = \langle ax, y \rangle + \langle bz, y \rangle \) (linearity);
2. \( \langle x, y \rangle = \overline{\langle y, x \rangle} \) (skew-symmetry);
3. \( \langle x, x \rangle \geq 0 \) with equality if and only if \( x = 0 \) (positive definiteness).

The inner product of \( \mathcal{H} \) induces a norm \( |x| := \sqrt{\langle x, x \rangle} \) for any vector \( x \in \mathcal{H} \) and a distance function \( d(x,y) := |x - y| \) between any two vectors \( x,y \in \mathcal{H} \). It is a routine exercise to check that this distance function in fact makes \( \mathcal{H} \) into a metric space, and that \( \mathcal{H} \) is complete with respect to this metric.

1.1.2 Notation and composite systems

It is convenient to use Dirac notation to represent quantum systems. A quantum system, as we shall see, is a vector in some state space \( \mathcal{H} \). We call such a vector a state vector. In Dirac notation, a vector is written as \( | \phi \rangle \in \mathcal{H} \), which we read as \textit{ket phi}. The conjugate transpose of \( | \phi \rangle \) is written \( \langle \phi | := | \phi \rangle^\dagger \), which we read as \textit{bra phi}. This notation is attractive because the product \( \langle \psi | \phi \rangle \) is in fact equal to \( \langle \psi | \phi \rangle \), the inner product of \( | \psi \rangle \) and \( | \phi \rangle \).

We use the notation \( | \phi \rangle \langle \psi | \) to represent the outer product of \( | \phi \rangle \) and \( | \psi \rangle \). The outer product is a linear transform \( | \phi \rangle \langle \psi | : \mathcal{H} \rightarrow \mathcal{H} \) that acts by

\[
| \phi \rangle \langle \psi | (| \theta \rangle) := \langle \psi | \theta \rangle | \phi \rangle.
\]

For instance, if \( | \phi \rangle \) is a unit vector, \( | \phi \rangle \langle \phi | \) is a rank one linear transform representing projection onto \( | \phi \rangle \).

Finally, we will use \( | \phi \rangle \otimes | \psi \rangle \) to denote the tensor product of \( | \phi \rangle \) and \( | \psi \rangle \). Suppose \( \mathcal{H}_1 \) and \( \mathcal{H}_2 \) are Hilbert spaces. Then \( \mathcal{H}_1 \otimes \mathcal{H}_2 \) is also a Hilbert space, of dimension \( \dim(\mathcal{H}_1) \cdot \dim(\mathcal{H}_2) \). The tensor product is a map \( \otimes : \mathcal{H}_1 \times \mathcal{H}_2 \rightarrow \mathcal{H}_1 \otimes \mathcal{H}_2 \) where we have the following identities for any \( | \phi_1 \rangle, | \phi_2 \rangle, | \phi \rangle \in \mathcal{H}_1 \), \( | \psi_1 \rangle, | \psi_2 \rangle, | \psi \rangle \in \mathcal{H}_2 \), and \( c \in \mathbb{C} \):
1.1. Information storage

1. \((|\phi_1 \rangle + |\phi_2 \rangle) \otimes |\psi \rangle = |\phi_1 \rangle \otimes |\psi \rangle + |\phi_2 \rangle \otimes |\psi \rangle\);

2. \(|\phi \rangle \otimes (|\psi_1 \rangle + |\psi_2 \rangle) = |\phi \rangle \otimes |\psi_1 \rangle + |\phi \rangle \otimes |\psi_2 \rangle\);

3. \(c |\phi \rangle \otimes |\psi \rangle = |\phi \rangle \otimes c |\psi \rangle = c (|\phi \rangle \otimes |\psi \rangle)\).

The inner product of a tensor product of two Hilbert spaces is the product of the inner products of the tensor factors; in symbols, we have

\[ \langle (|\phi_1 \rangle \otimes |\psi_1 \rangle), (|\phi_2 \rangle \otimes |\psi_2 \rangle) \rangle = \langle \phi_1 | \phi_2 \rangle \cdot \langle \psi_1 | \psi_2 \rangle. \]

Instead of \(|\phi \rangle \otimes |\psi \rangle\), we often write \(|\phi \rangle |\psi \rangle\) or even \(|\phi \psi \rangle\) if this is unambiguous. Of course it is possible to take the tensor product of more than two Hilbert spaces; it is routine to verify that the order in which we take the products does not matter, i.e. that \(\mathcal{H}_1 \otimes (\mathcal{H}_2 \otimes \mathcal{H}_3) = (\mathcal{H}_1 \otimes \mathcal{H}_2) \otimes \mathcal{H}_3\).

The tensor product is extremely important in the mathematical formulation of quantum mechanics because the composite of multiple quantum systems is the tensor product of the constituent systems.

**Postulate 2. (Composite systems)**

If \(\mathcal{H}_1, \ldots, \mathcal{H}_k\) are quantum state spaces, then the composite system of these spaces is \(\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_k\). If \(\mathcal{H}_i\) is in state \(|\phi_i \rangle\) for all \(1 \leq i \leq k\), then \(\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_k\) is in state \(|\phi_1 \rangle \otimes \cdots \otimes |\phi_k \rangle\).

1.1.3 State vectors

The **bit** is the basic unit of information for classical computers. A bit can have one of two values, 0 or 1. Any natural number can be stored as a series of bits by writing it in binary; e.g. \(5 = 101\). We see that it takes \(\log(N)\) bits to represent the number \(N \in \mathbb{N}\).

The **qubit** is the quantum analogue of the bit. Like the bit, a qubit, which we regard as a state vector \(|\phi \rangle\) in the Hilbert space \(\mathbb{C}^2\), can take on the classical values \(|0 \rangle\) and \(|1 \rangle\). However, a qubit may also be in a superposition of these two states:

\[ |\phi \rangle = \alpha |0 \rangle + \beta |1 \rangle. \]

Here we consider \(\{ |0 \rangle, |1 \rangle \}\) to be an orthonormal basis of \(\mathbb{C}^2\). Which values of \(\alpha\) and \(\beta\) are permissible? To answer this we need another postulate.

**Postulate 3. (State vectors)**

A **state vector** is a unit vector in some state space.\(^1\)

\(^1\)If two state vectors differ only by a global phase, i.e. if \(|\phi \rangle = e^{i\theta} |\psi \rangle\), then, as we shall see, there is no way to distinguish \(|\phi \rangle\) and \(|\psi \rangle\) via a measurement. While it might therefore be tempting to consider a state vector to be an equivalence class of unit vectors that differ by a global phase, this would cause some problems with our definition of a composite state. We will consider a state vector to be a single vector.
Thus we see that in $|\phi\rangle$ above we have $|\alpha|^2 + |\beta|^2 = 1$. We may also consider a state vector representing multiple qubits, for instance:

$$|\psi\rangle = \alpha |00\rangle + \beta |01\rangle + \gamma |10\rangle + \delta |11\rangle.$$ 

In this case $|\psi\rangle \in \mathbb{C}^4 \simeq \mathbb{C}^2 \otimes \mathbb{C}^2$, and again we have $|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$. It is often convenient to look at parts of our qubit systems, and for this purpose the notion of a register is convenient.

**Definition 1.1.** A register of a state vector involving multiple qubits is some subset of all of the qubits.

The notation $|\phi_1\rangle|\phi_2\rangle$ stresses that we are viewing a state vector as comprising two registers. So for instance we write $|\psi\rangle = \alpha |00\rangle + \beta |10\rangle + \gamma |00\rangle + \delta |10\rangle$ to represent $|\psi\rangle$ as comprising two single qubit registers.

Just as we can encode a natural number on a classical computer by writing its binary representation in bits, we can encode a number on a quantum computer by writing its binary representation in qubits. For instance, $|5\rangle = |101\rangle = |1\rangle |0\rangle |1\rangle$. We can encode any finite set of numbers in qubits, so although we assume our quantum computer stores its information in qubits, we have no problem working in the Hilbert space $\mathbb{C}^N$ whose orthonormal basis vectors we will call $|0\rangle, |1\rangle, \ldots, |N-1\rangle$. In fact, any finite set $X$ can be encoded in qubits by some map $X \to \{0, \ldots, |X| - 1\}$, so we will also feel comfortable working in the Hilbert space $\mathbb{C}^X \simeq \mathbb{C}^{|X|}$ whose orthonormal basis vectors are $|x\rangle$ for $x \in X$. Sometimes we use the notation $\mathbb{C}^{|x\rangle}$ to mean the subspace of $\mathbb{C}^X$ spanned by $|x\rangle$.

### 1.2 Information processing

In the last section, we explained how there is a significant difference in the way classical and quantum computers store information (bits versus qubits). There is a similarly significant difference in information processing between the two kinds of computers. In a classical computer, bits can be rewritten at will; however, because the evolution of a quantum state is symmetric with respect to time, in a quantum computer all operations on qubits must be reversible. In particular, we have the following postulate:

**Postulate 4. (State evolution)**

Quantum systems evolve over time by unitary transforms on their Hilbert spaces, and any unitary transform represents a valid evolution of a quantum system. In other words, if a quantum system is in state $|\phi\rangle \in \mathcal{H}$ at time $t$ and at a later time $t'$ is in state $|\phi'\rangle$, then there exists some unitary transform $U$ such that $|\phi'\rangle = U |\phi\rangle$.

A unitary transform on a Hilbert space $\mathcal{H}$ is a bijective linear transformation $U : \mathcal{H} \to \mathcal{H}$ that preserves inner products:

$$\langle \psi, \phi \rangle = \langle U \psi, U \phi \rangle \text{ for all } |\phi\rangle, |\psi\rangle \in \mathcal{H}.$$
1.2. Information processing

Equivalently, $U$ is a bijective linear transformation such that $UU^\dagger = I$, where $I$ is the identity map on $\mathcal{H}$. Or equivalently again, $U$ maps an orthonormal basis of $\mathcal{H}$ to another orthonormal basis of $\mathcal{H}$.

The tensor product of two unitary transforms $U_1 : \mathcal{H}_1 \rightarrow \mathcal{H}_1$ and $U_2 : \mathcal{H}_2 \rightarrow \mathcal{H}_2$, which we write $U_1 \otimes U_2$, operates on a vector $|\psi\rangle \otimes |\phi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$ in the way one would expect:

$$(U_1 \otimes U_2)(|\psi\rangle \otimes |\phi\rangle) = U_1 |\psi\rangle \otimes U_2 |\phi\rangle.$$ 

An example of a unitary transform that is extremely important for quantum computing is the Hadamard transform, $H : \mathbb{C}^2 \rightarrow \mathbb{C}^2$. The Hadamard transform can be written in matrix form as

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

with respect to the orthonormal basis $\{|0\rangle, |1\rangle\}$ of $\mathbb{C}^2$. So $H |0\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$ and $H |1\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$. These states are themselves so important that we give them special names:

$$|+\rangle := \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle),$$

$$|−\rangle := \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle).$$

Another example of a basic unitary transform on $\mathbb{C}^2$ is

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

We have $X |0\rangle = |1\rangle$ and $X |1\rangle = |0\rangle$. Because it flips the value of a qubit, we call $X$ the quantum NOT transform. A unitary transform related to NOT but which acts on $\mathbb{C}^4$ is:

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

This matrix representation of CNOT assumes our basis is $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. Hence we have CNOT $|00\rangle = |00\rangle$, CNOT $|01\rangle = |01\rangle$, CNOT $|10\rangle = |11\rangle$, and CNOT $|11\rangle = |10\rangle$. As we can see, CNOT flips the second qubit of the tensor product if the first qubit is $|1\rangle$, and acts as the identity if the first qubit is $|0\rangle$. Thus we call CNOT the controlled NOT transform.

There exists a set of four unitary transforms including $H$ and CNOT called a set of universal quantum gates that can approximate any unitary transform arbitrarily well.\footnote{These two transforms along with the phase and $\pi/8$ transforms suffice; see [15], section 4.5, for a precise explanation. The terms “universal quantum gate” and “circuit complexity” come from a model of quantum computers as circuits.} That is, we can get a very good approximation of an arbitrary transform
unitary $U$ with a finite product of these simpler transforms. We assume our quantum computer at least has the ability to perform the universal quantum gates on its qubits. Because the product of two unitary transforms is again unitary, the application of a series of transforms ($U_1$ followed by $U_2$ and so on up to $U_n$) could just as easily be seen as the application of a single unitary $U := U_n U_{n-1} \cdots U_1$. However, we are interested in computational efficiency, so we care about how many simple transforms are needed to execute our algorithm.

**Definition 1.2.** The *circuit complexity* of a quantum algorithm is the number of universal quantum gates required to approximate the unitary transforms used in the algorithm.

Whether we are analyzing classical or quantum algorithms, an algorithm is considered efficient if it takes time polynomial in the size of the input. Since our inputs are stored as a series of qubits, an input of $N \in \mathbb{N}$ has size $\log(N)$. Thus we are interested in algorithms that take time $f(N)$ polynomial in the logarithm of $N$; we write $f = \text{poly}(\log(N))$ in such a case. We want the circuit complexity of our algorithm to be $\text{poly}(\log(N))$ as a function of the input size.

### 1.3 Information retrieval

Information retrieval also differs greatly between classical computers and quantum computers. Rarely do we even consider our ability to retrieve the information stored in a classical computer: we are allowed to access the true value of any of our bits at any time, and looking up a bit does not affect its value. On the other hand, when we measure a qubit, we only receive classical information (a value of 0 or 1) even if the qubit was in a superposition between $|0\rangle$ and $|1\rangle$, and in measuring the qubit we collapse it to the classical state we measured. Importantly, while the unitary operations we considered in the previous section were all reversible, measurement of a quantum state is an irreversible process.

**Postulate 5.** *Measurement*

Suppose $\{M_m\}$ is a set of linear transforms acting on a state space $\mathcal{H}$ such that $\sum_m M_m = I$. Then we call these $M_m$ *measurement operators*, and they allow us to measure a system $|\phi\rangle \in \mathcal{H}$. The probability that we measure the value $m$ is given by

$$P(m) = \langle \phi | M_m^\dagger M_m | \phi \rangle,$$

and if we do measure the value $m$, the state of the system post-measurement is

$$|M_m |\phi\rangle / |M_m |\phi\rangle|.$$

(Note that the requirement $\sum_m M_m = I$, called *completeness*, ensures that the laws of probability are obeyed.)

The most important example of measurement is measurement in the computational basis:
Definition 1.3. Let $|\phi\rangle \in \mathbb{C}^X$ be some state vector. To measure $|\phi\rangle$ in the computational basis is to measure $|\phi\rangle$ with the set of measurement operators being the set $\{|x\rangle \langle x|\}_{x \in X}$. Measurement in the computational basis yields some value $x \in X$.

For example, if we measure $|+\rangle$ in the computational basis, we obtain value 0 half the time, and we obtain the value 1 half the time. After measurement, the state becomes $|0\rangle$ or $|1\rangle$ depending on which value we measured. There are many more sophisticated kinds of measurements, but in this thesis we will need only measurement in the computational basis and partial measurement, which we explain with the next definition. Thus, whenever we say we measure some state vector, we mean that we measure it in the computational basis.

Definition 1.4. Suppose $|\psi\rangle |\phi\rangle$ is some vector in $\mathcal{H} \otimes \mathbb{C}^X$. To measure the second register of $|\psi\rangle |\phi\rangle$ in the computational basis is to measure $|\psi\rangle |\phi\rangle$ with the set of measurement operators being the set $\{(I \otimes |x\rangle \langle x|)\}_{x \in X}$, where $I$ is the identity on $\mathcal{H}$. This measurement yields some value $x \in X$. Measurement of the first register is defined analogously.

Note that the state after measurement always projects to a unit vector. For example, let $|\phi\rangle = \frac{1}{\sqrt{2}} (|0\rangle |0\rangle + |1\rangle |1\rangle)$. Then, when we measure the second register of $|\phi\rangle$, we obtain 0 half the time and 1 half the time. If we measure 0, the state becomes $|0\rangle |0\rangle$, whereas if we measure 1, the state becomes $|1\rangle |1\rangle$. Thus we see that a measurement of the second register has affected the value of the first register. This phenomenon, known as quantum entanglement, is important in explaining how quantum computers might offer a computational speed-up from their classical analogues.

1.4 The quantum Fourier transform

One unitary transform in particular, the quantum discrete Fourier transform, will be a key tool in our development of a quantum algorithm. There are only a few categories of quantum algorithms that perform better than the best known classical algorithms for the same problem; algorithms based on the quantum Fourier transform make up one of these categories.\(^3\)

Definition 1.5. The quantum Fourier transform, $\mathcal{F}_N : \mathbb{C}^N \to \mathbb{C}^N$, acts on a basis of $\mathbb{C}^N$ by

$$\mathcal{F}_N |j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{jk} |k\rangle,$$

where $\omega := e^{2\pi i/N}$ is a primitive $N$th root of unity.

\(^3\)Examples of other classes include quantum search algorithms, such as Grover’s algorithm, and quantum annealing algorithms.
We can easily verify that the quantum Fourier transform is unitary. First we claim that the inverse of the quantum Fourier transform acts as

$$F_N^{-1} |j⟩ := \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{-jk} |k⟩.$$ 

Recall the following basic fact about roots of unity:

**Fact 1.6.** Let $ζ$ be a primitive $m$th root of unity, with $m > 1$, and let $r$ be some integer. Then we have

$$\frac{1}{m} \sum_{k=0}^{m-1} ζ^{rk} = \begin{cases} 1 & r = 0; \\ 0 & r ≠ 0. \end{cases}$$

Straightforward computation, along with this fact, shows that the above formula for the inverse Fourier transform is correct:

$$F_N^{-1} F_N |j⟩ = \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} \omega^{jl} \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{-lk} |k⟩$$

$$= \frac{1}{N} \sum_{l=0}^{N-1} \sum_{k=0}^{N-1} \omega^{l(j-k)} |k⟩$$

$$= |j⟩.$$ 

Finally, to conclude that $F_N$ is unitary, note that the formula for $F_N^{-1}$ makes explicit that $F_N^† = F_N^{-1}$, and so $F_N F_N^† = I$.

Quantum algorithms based on the quantum Fourier transform depend crucially on the fact that it can be efficiently implemented by a quantum computer. Currently, the best implementations of the quantum Fourier transform have a circuit complexity of only $O(\log N \log \log N)$ [11]. It is difficult to explain exactly why the Fourier transform is so successful in a wide range of applications, but, roughly speaking, the Fourier transform gives information about the periodicity of a function. Worth noting is that the quantum Fourier transform of size two is just the Hadamard transform: $F_2 = H$.

### 1.5 Oracle problems

We have now developed enough background in quantum computing to present an example of a quantum algorithm: Deutsch’s algorithm. Deutsch’s algorithm solves an oracle problem.

**Definition 1.7.** In an oracle problem, we are given access to a certain unitary transform, $O_f: \mathbb{C}X \otimes \mathbb{C}Y \rightarrow \mathbb{C}X \otimes \mathbb{C}Y$, called an oracle, that acts on a basis of $\mathbb{C}X \otimes \mathbb{C}Y$ by

$$O_f |i⟩ |r⟩ = |i⟩ |f(i) + r⟩,$$
where \( f \) is some unknown function from \( X \) to \( Y \). (Here, in order that \( O_f \) be unitary, we assume addition makes sense in \( Y \); that is, we assume \( Y \) is an abelian group.)

The function \( f \) is guaranteed to be from some set of functions \( \mathcal{F} = \mathcal{F}_1 \cup \cdots \cup \mathcal{F}_k \), and \textit{a priori} we assume some probability distribution over the \( f \in \mathcal{F} \). Our goal is to identify the family \( \mathcal{F}_i \) that our function \( f \) belongs to with as few applications of the oracle as possible.

We call the first register, \( \mathbb{C}X \), the \textit{query register}, and the second register, \( \mathbb{C}Y \), the \textit{response register}. Two registers are required because we need the action of the oracle to be reversible. Although this description of an oracle problem may make it seem like a contrived concept, in fact such problems arise quite naturally in theoretical computer science. We are not really interested in probing some black box for its hidden information. Rather, we think of the oracle as a computationally expensive subroutine whose behavior we cannot predict except by evaluating it on certain inputs. We can therefore see why we want to minimize the number of calls to the subroutine.

In analyzing the efficiency of an algorithm that solves an oracle problem, we must consider both the circuit complexity and query complexity.

\textbf{Definition 1.8.} The \textit{query complexity} of an algorithm that solves an oracle problem is the number of applications of the oracle it requires.

We are looking for algorithms that solve problems in a probabilistic sense. That is, we say that an algorithm solves a problem if it yields the desired result with probability arbitrarily close to 1. Note that if an algorithm efficiently yields the desired result with probability greater than \( c \), where \( c > 1/2 \) is some fixed constant, then in fact this algorithm solves the problem efficiently. This follows from a basic statistical fact: if we have a coin that is biased towards either heads or tails, with only a constant number of flips we are able to determine the bias with probability exponentially close to 1 by concluding that it is biased towards the side it lands on a majority of the time.

Deutsch’s problem, unlike oracle problems generally, is rather contrived. The oracle in Deutsch’s problem hides two bits \( x_0 \) and \( x_1 \) from us and we are asked to determine the binary sum \( x_0 + x_1 \). It is clear that a classical computer requires two queries to this oracle to determine \( x_0 + x_1 \) because it needs to know both bits to know the sum; in particular, with only a single oracle query, a classical computer has a probability \( 1/2 \) of determining the sum because the sum takes the values 0 or 1 with equal probability independent of the value of one of the hidden bits. Deutsch’s quantum algorithm solves Deutsch’s problem with only one query. We present Deutsch’s problem here because historically it was one of the first problems to show that quantum computers could be computationally more powerful than their classical analogues, because its solution is simple to explain, and especially because it is a special case of the Hidden Subgroup Problem, a problem which we will fully define in the next chapter and which is the focus of this thesis.

\textbf{Definition 1.9.} Let \( O_{x_0,x_1} : \mathbb{C}Z_2 \otimes \mathbb{C}Z_2 \rightarrow \mathbb{C}Z_2 \otimes \mathbb{C}Z_2 \), with \( x_0, x_1 \in \mathbb{Z}_2 \) act by

\[
O_{x_0,x_1} |i\rangle |r\rangle = |i\rangle |x_i + r\rangle .
\]
Deutsch’s problem is to determine the sum \( x_0 + x_1 \) (mod 2).

**Theorem 1.10.** There is a quantum algorithm that solves Deutsch’s problem with a single application of the oracle.

**Proof:** We prove this theorem by demonstrating the algorithm. Deutsch’s algorithm, first devised in [5], proceeds as follows:

1. Prepare the state \( |\phi\rangle := |0\rangle|1\rangle \in \mathbb{CZ}_2 \otimes \mathbb{CZ}_2 \). Note that \( |\phi\rangle = (I \otimes X)|0\rangle|0\rangle \), so we could have as easily started with \( |0\ldots0\rangle \), which is a state we assume our quantum computer can prepare.

2. Apply \( H \otimes H \) to yield \((H \otimes H)|\phi\rangle = |+\rangle|\rangle\).

3. Apply \( O_{x_0, x_1} \) to yield

\[
O_{x_0, x_1}(H \otimes H)|\phi\rangle = \frac{1}{2} (|0\rangle|x_0\rangle - |0\rangle|x_0 + 1\rangle + |1\rangle|x_1\rangle - |1\rangle|x_1 + 1\rangle).
\]

If \( x_0 = 0 \), then \( |0\rangle|x_0\rangle - |0\rangle|x_0 + 1\rangle = |0\rangle|0\rangle - |0\rangle|1\rangle = \sqrt{2}|0\rangle|\rangle\), and if \( x_0 = 1 \), then \( |0\rangle|x_0\rangle - |0\rangle|x_0 + 1\rangle = -\sqrt{2}|0\rangle|\rangle\). The value of \(|1\rangle|x_1\rangle - |1\rangle|x_1 + 1\rangle\) depends on \( x_1 \) similarly. Therefore we may rewrite this state as

\[
O_{x_0, x_1}(H \otimes H)|\phi\rangle = \frac{1}{\sqrt{2}} ((-1)^{x_0} |0\rangle|\rangle + (-1)^{x_1} |1\rangle|\rangle) \\
= \frac{(-1)^{x_0}}{\sqrt{2}} (|0\rangle|\rangle + (-1)^{x_1 - x_0} |1\rangle|\rangle).
\]

4. Apply \((H \otimes I)\) to yield

\[
(H \otimes I)O_{x_0, x_1}(H \otimes H)|\phi\rangle = \frac{(-1)^{x_0}}{\sqrt{2}} ((+\rangle|\rangle + (-1)^{x_1 - x_0} |\rangle|\rangle) \\
= \frac{(-1)^{x_0}}{\sqrt{2}} ((1 + (-1)^{x_1 - x_0}) |0\rangle|\rangle + (1 - (-1)^{x_1 - x_0}) |1\rangle|\rangle).
\]

5. Measure the first register. If \( x_1 = x_0 \), then our state is \(|0\rangle|\rangle\), so we measure 0 with probability 1. On the other hand, if \( x_1 \neq x_0 \), then our state is \(|1\rangle|\rangle\), so we measure 1 with probability 1. Either way, we yield the sum \( x_0 + x_1 \). □

Since Deutsch’s algorithm invokes the oracle only once, we call it a *single-query* algorithm. To solve more sophisticated problems, we will consider *multi-query* algorithms that invoke the oracle multiple times, either in series or parallel. Also, as explained earlier, we do not require that our algorithm succeed with probability 1 as Deutsch’s algorithm does. Despite these differences from more advanced algorithms, Deutsch’s algorithm showcases the general structure of any quantum algorithm that solves an oracle problem: prepare a certain state via unitary transforms applied to \(|0\ldots0\rangle\), apply the oracle to this state, apply more unitary transforms to the post-oracle state, and finally measure the state to obtain the desired result.
Chapter 2

The Hidden Subgroup Problem

In this chapter we explore the group structure underlying problems in quantum computing, like Deutsch’s problem, and see how, as in the case of Deutsch’s algorithm, the Fourier transform is instrumental in attacking such problems. The Hidden Subgroup Problem provides a unified framework for describing many of the known quantum algorithms and for suggesting problems that may admit of some quantum solution. We recount the state of the art of the Hidden Subgroup Problem here so that in the next chapter we may build on existing research and develop an algorithm for a particular case of the problem, relative to certain assumptions about the feasibility of another proposed algorithm.

2.1 History and overview

The Hidden Subgroup Problem (HSP) was first introduced by Brassard and Høyer [1] as a generalization of Simon’s problem, another early quantum problem. Later it was seen that not only Simon’s algorithm, but also Deutsch’s algorithm and the most remarkable known quantum algorithm, Shor’s algorithm for factorizing integers, all solve special cases of the HSP. Indeed, the key quantum component of Shor’s algorithm,\footnote{Shor’s algorithm [22] has two major components: a classical number-theoretic reduction of factoring to finding the order of an element of the multiplicative group \( \mathbb{Z}_N^* \), and a quantum algorithm for order-finding.} the period-finding subroutine, reduces to the HSP for cyclic groups. The HSP offers an elegant generalization of the task of determining the discrete period of some unknown function to determining the “period” of an unknown function that is “periodic” over a group; this is why we might expect the Fourier transform to be a crucial tool. Contemporary research has attempted to find efficient solutions to the HSP in the case of non-abelian groups. Several classes of non-abelian groups have HSPs that are equivalent to seemingly-unrelated, outstanding problems in computer science, such as the graph isomorphism problem. No general solution to the HSP exists, and the problem remains an area of active research because it encapsulates many independently interesting problems and lies on the border of what might be computationally feasible with a quantum computer.
2.2 Statement of the problem

Definition 2.1. Let $G$ be a finite group, $X$ a finite set, $H$ a subgroup of $G$, and $f: G \rightarrow X$ a function such that $f(g) = f(g')$ if only if $g' \in gH$. In other words, $f$ is constant and distinct on left cosets of $H$. We say that $f$ hides $H$. The Hidden Subgroup Problem (HSP) is, given such an $f$ (and therefore knowledge of $G$ and $X$), find the subgroup that it hides.

The HSP can be realized as an oracle problem in the following manner. Let $L$ be the set of subgroups of $G$ we are trying to distinguish among, and let $F$ be the set of all functions $f$ for which there exists some subgroup $H \in L$ such that $f$ hides $H$. For each $f \in F$, define $O_f : \mathbb{C}G \otimes \mathbb{C}X \rightarrow \mathbb{C}G \otimes \mathbb{C}X$ by

$$O_f |g\rangle|r\rangle = |g\rangle|f(g) + r\rangle.$$  

Then the HSP asks us to determine the subgroup $H$ hidden by the function $f$ with as few queries to the oracle $O_f$ as possible. Usually we assume that $L$ is the set of all subsets of $G$, and sometimes we adjust the probability distribution over $F$ to make each subgroup equally likely to be chosen. We often ignore the particular function that hides our subgroup and write $O_H := O_f$, where $f$ is an arbitrary function that hides $H$.

Deutsch’s problem is an instance of the HSP. In particular, if we set $G = \mathbb{Z}_2$, $X = \mathbb{Z}_2$, and the hidden subgroup $H$ to be either the full group $\mathbb{Z}_2$ or the trivial group, we recover Deutsch’s problem. The case $H = \mathbb{Z}_2$ corresponds to a sum $x_0 + x_1 = 0$, and the case of $H$ trivial corresponds to a sum $x_0 + x_1 = 1$.

Shor’s problem of finding the order of some element $x$ of the multiplicative group $\mathbb{Z}_N^*$ is also an instance of the HSP. Suppose we are given access to the function $f: \mathbb{Z}_N \rightarrow \mathbb{Z}_N^*$ defined by $f(a) = x^a \pmod{N}$. Then, if the order of $x$ is $r$, we have $f(a + r) = f(a)$ for all $a \in \mathbb{Z}_N$. So with $G = \mathbb{Z}_N$ and $X = \mathbb{Z}_N^*$, we see that the function $f$ hides the subgroup $\langle r \rangle$.

2.3 The abelian HSP

The abelian HSP has an efficient quantum solution. We will now work through a solution for a special case of the abelian HSP, namely, the HSP for cyclic groups. Not only is this algorithm quite elegant, but it also offers a good example of the use of entanglement in quantum computing and has some structural similarity to the HSP algorithm for a non-abelian group we will develop in the next chapter.

Theorem 2.2. There exists an efficient quantum algorithm for solving the HSP for the cyclic group $\mathbb{Z}_N$.

Proof: We prove this theorem by demonstrating the algorithm. Our presentation follows Lomont [13], section 3.4. First, realize that the subgroups of $G$ are $H_d := \langle d \rangle$ for those $d \in \mathbb{Z}_N$ such that $d \mid N$. Suppose our hidden subgroup is $H_d$ and then let $M := N/d$. The algorithm repeats the following procedure $\kappa$ times, for some constant $\kappa$, obtaining the samples $t_1, t_2, \ldots, t_\kappa$ as the result of the final measurement in step 6:
2.3. The abelian HSP

1. Prepare the state $|0\rangle |0\rangle \in \mathbb{C}Z_N \otimes \mathbb{C}Z_N$.

2. Apply $(\mathcal{F}_N \otimes I)$ to yield the equal superposition state $|\phi\rangle := \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle |0\rangle$.

3. Apply $\mathcal{O}_{H_d}$ to yield

$$\mathcal{O}_{H_d}(\mathcal{F}_N \otimes I) = \sum_{j=0}^{N-1} |j\rangle |f_{H_d}(j)\rangle .$$

4. Measure the second register. This obtains some value $f_{H_d}(j_0)$, and, thanks to quantum entanglement, collapses the first register into an equal superposition over elements in the coset $j_0 + H_d$. Call the resulting vector $|j_0 + H_d\rangle$:

$$|j_0 + H_d\rangle = \frac{1}{\sqrt{M}} \sum_{h \in H_d} |j_0 + h\rangle = \frac{1}{\sqrt{M}} \sum_{s=0}^{M-1} |j_0 + sd\rangle .$$

5. Apply $\mathcal{F}_N$ again, to give the state

$$\mathcal{F}_N |j_0 + H_d\rangle = \frac{1}{\sqrt{M}} \sum_{s=0}^{M-1} \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i (j_0 + sd)k/N} |k\rangle$$

$$= \frac{1}{\sqrt{MN}} \sum_{k=0}^{N-1} e^{2\pi ij_0k/M} |k\rangle \sum_{s=0}^{M-1} e^{2\pi isdk/N} .$$

But then note that, for some fixed $k$,

$$\sum_{s=0}^{M-1} e^{2\pi isdk/N} = \sum_{s=0}^{M-1} (e^{2\pi i/M})^s = \left\{ \begin{array}{ll} 0 & M \nmid k; \\
M & M \mid k. \end{array} \right.$$ 

So only those $|k\rangle$ that are multiples of $M$ have nonzero amplitudes, and thus our state is in fact

$$\mathcal{F}_N |j_0 + H_d\rangle = \frac{1}{\sqrt{d}} \sum_{t=0}^{d-1} e^{2\pi ij_0tM/N} |tM\rangle .$$

6. Measure the state vector. We obtain a random element of $\{0, M, \ldots, (d-1)M\}$ with uniform probability.

After repeating the above $\kappa$ times, we are left with samples $t_1, \ldots, t_\kappa$, which are all random multiples of $M$. It is a well-known fact from number theory that, with high probability, the greatest common divisor of a few random numbers in $\{0, \ldots, d-1\}$ is 1; for instance, see Lomont’s demonstration in [13], Appendix E. In fact, Lomont
shows $\kappa = 8$ suffices to guarantee that this algorithm succeeds with probability greater than $1/2$. So we take $\gcd(t_1, \ldots, t_8)$ and with high probability get $M$; we then return $N/M = d$ and have solved the HSP for $\mathbb{Z}_N$. □

This algorithm can be generalized to work for an arbitrary finite abelian group $G$. The following classical result from group theory offers at least a hint of why that is so:

**Theorem 2.3.** (Fundamental Theorem of Finite Abelian Groups, see [6], §5.2)
Any finite abelian group $G$ is isomorphic to the direct product of a finite number of cyclic groups.

A general solution to the abelian HSP will break $G$ into its cyclic group factors and apply the above algorithm to these separate factors. Making this idea more precise is beyond the scope of this chapter; see [13] for the details.

### 2.4 The non-abelian HSP

There is no known general polynomial-time solution to the non-abelian HSP, but the HSP has been efficiently solved for certain classes of non-abelian groups such as “almost abelian” groups [24], “nearly Hamiltonian” groups [9], the wreath product $\mathbb{Z}_2^n \wr \mathbb{Z}_2$ [18], certain groups of the form $\mathbb{Z}_{p^k}^r \rtimes \mathbb{Z}_p$ [19], groups of the form $\mathbb{Z}_p^r \rtimes \mathbb{Z}_p$ [3], and so on.

Almost all known algorithms for the non-abelian HSP emulate the abelian HSP algorithm presented in the last section by following the so-called Standard Method. The Standard Method proceeds as follows:

1. Prepare an equal superposition over group elements in the first register and $|0\rangle$ in the response register:

   $$|\Phi\rangle := \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |0\rangle.$$ 

2. Apply $O_{f_H}$:

   $$O_{f_H} |\Phi\rangle = \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |f_H(g)\rangle.$$ 

3. Measure the second register in order to collapse the first into $|gH\rangle$, an equal superposition over some coset $gH$ for a random $g \in G$:

   $$|gH\rangle := \frac{1}{\sqrt{|H|}} \sum_{h \in H} |gh\rangle.$$ 

Then repeat the above steps $\kappa$ times in parallel, obtaining samples $|g_1H\rangle, \ldots, |g_\kappa H\rangle$, and attempt to determine $H$ from these coset states. Again, almost all HSP research has focused on the Standard Method, but there is no proof that this approach is
optimal. At any rate, the HSP algorithm for a non-abelian group we develop in the next chapter will employ the Standard Method.

The HSP for the symmetric group and the HSP for the dihedral group are of particular interest because of their connections to other problems in computer science. An efficient solution to the symmetric group HSP would imply a solution to the notorious graph isomorphism problem [7]. The graph isomorphism problem is one of only a handful of NP problems, like integer factorization, that are neither believed to be in P nor believed to be NP-complete.\(^2\) Thus there is some hope that an efficient quantum algorithm for graph isomorphism exists because the problem is “well-structured” in a way that may be particularly suited to quantum computing. Nevertheless, researchers have so far been unable to find an efficient algorithm for the symmetric group HSP.

An efficient solution to the dihedral HSP would imply a solution to a class of shortest vector problems in lattices [16]. The supposed intractability of these shortest vector problems is the basis for certain proposed cryptography systems; thus, there is considerable practical interest in developing a quantum dihedral HSP algorithm. As in the case of the symmetric group, there is no known efficient solution to the dihedral HSP. However, there are many partial results. Regev [16] has shown that an efficient, approximate solution to the subset sum problem (defined formally in section 3.6) implies an efficient solution to the dihedral HSP. Kuperburg [12] discovered a subexponential, but not polynomial, time algorithm for the dihedral HSP. Bacon et al. [3] showed that, under the assumption of the Standard Method, a general measurement technique for distinguishing quantum states, the so-called Pretty Good Measurement, is optimal for the dihedral HSP. Further, in [4] they extend the results of Regev and show that the ability to implement the Pretty Good Measurement for the dihedral HSP is equivalent to the ability to quantum sample solutions to the subset sum problem.

In the next chapter, we build on the work of Bacon et al. and show that their proposed algorithm for the dihedral HSP, which assumes a kind of quantum subset sum solver, in fact applies to a much broader class of groups. The dihedral group $D_n$ is the group of symmetries of a regular $n$-gon, but can also be envisioned more abstractly as a semidirect product $\mathbb{Z}_N \rtimes \mathbb{Z}_2$. The algorithm in the next chapter applies to arbitrary semidirect products of a cyclic group by a prime cyclic group. The fact that the dihedral algorithm of Bacon et al. [4] readily generalizes to $\mathbb{Z}_N \rtimes \mathbb{Z}_p$ (for fixed $p$) may suggest that the assumption of an efficient quantum subset sum solver is too strong, and that no such algorithm exists. Regardless, the connection between the HSP for $\mathbb{Z}_N \rtimes \mathbb{Z}_2$ and for $\mathbb{Z}_N \rtimes \mathbb{Z}_p$ is worth demonstrating.

---

\(^2\)The complexity class P consists of decision problems that can be solved in polynomial time (i.e., those admitting what we have termed an “efficient” solution). The complexity class NP consists of decision problems whose solutions can be verified in polynomial time. NP-complete problems are, roughly, “the hardest problems in NP.” For a technical treatment of computational complexity, see [23], chapter 7.
Chapter 3

The HSP for $\mathbb{Z}_N \rtimes \mathbb{Z}_p$

In this chapter we present original research. The major theorem of this chapter is the following:

**Theorem 3.1.** If there is an efficient implementation of the Pretty Good Measurement for the dihedral HSP, there is an efficient quantum algorithm that solves the HSP for $\mathbb{Z}_N \rtimes \mathbb{Z}_p$, with $N$ any natural number and $p$ prime.

We prove this theorem by demonstrating such an algorithm. Ettinger and Høyer [8] have reduced the dihedral HSP to the problem of finding a hidden trivial subgroup or hidden order 2 subgroup. Bacon et. al. [3] generalized this reduction by reducing the HSP over $A \rtimes \mathbb{Z}_p$ to finding a hidden trivial subgroup or hidden cyclic order $p$ subgroup. They then apply the Pretty Good Measurement and discover efficient algorithms for several classes of semidirect product groups, including the metacyclic group $\mathbb{Z}_N \rtimes \mathbb{Z}_p$, provided that $N/p$ is poly($\log(N)$).

Here we consider the hidden subgroup problem on $\mathbb{Z}_N \rtimes \mathbb{Z}_p$ with $N/p$ super-polynomial in the logarithm of $N$. We generalize an algorithm presented in Asif Shakeel’s PhD thesis [21] for the dihedral group (which can be realized as $\mathbb{Z}_N \rtimes \mathbb{Z}_2$) to $\mathbb{Z}_N \rtimes \mathbb{Z}_p$. The dihedral algorithm Shakeel presents comes from another paper by Bacon et. al. [4] Our algorithm, as in the dihedral case, assumes that we can efficiently quantum sample solutions to the subset sum problem, which is known to be NP-complete. Bacon et. al. [4] show that being ample to implement the Pretty Good Measurement for the dihedral HSP implies the ability to quantum sample subset sum solutions.

### 3.1 The metacyclic group $\mathbb{Z}_N \rtimes \mathbb{Z}_p$

Let $p$ be a prime and $N$ any integer. The metacyclic group $G := \mathbb{Z}_N \rtimes \mathbb{Z}_p$ is presented as $\langle a, b \mid a^p = b^N = e, aba^{-1} = b^k \rangle$ for some $k$ such that $k^p = 1 \pmod{N}$. From now on we write $\mathbb{Z}_N \rtimes_k \mathbb{Z}_p$ to stress that the conjugation factor $k$ is necessary to precisely specify the group and that we always have knowledge of $k$. Note that $k = 1$ gives the direct product, an abelian group whose HSP has been efficiently solved, so we assume $k \neq 1$. Also, $k = -1$ is only possible for $p = 2$; in this case, we have the
dihedral group whose HSP is the subject of [21], so we assume further that \( k \neq -1 \). Of course, \( k \in \mathbb{Z}_N^* \).

We will write \( G = \{(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_N\} \), with multiplication given by

\[
(x', y') \cdot (x, y) = (x + x', k^x y' + y).
\]

(N.B.: In keeping with Shakeel’s presentation of the dihedral algorithm, we have chosen to write the pairs \( (x, y) \in G \) with the first coordinate corresponding to \( \mathbb{Z}_p \) and the second corresponding to \( \mathbb{Z}_N \), even though the notation \( \mathbb{Z}_N \rtimes_k \mathbb{Z}_p \) might suggest the opposite is more natural.)

Following the notation of Bacon et. al., for convenience we define \( \Phi^j \in \mathbb{Z}_N \) as

\[
\Phi^j := \sum_{i=0}^{j-1} k^i.
\]

Define the subgroups

\[
H_l := \langle (1, l) \rangle = \{(0, 0), (1, l), \ldots, (p-1, (k^{p-2} + \cdots + 1)l)\} = \{(a, \Phi^l)\}_{a=0}^{p-1},
\]

for \( l \in \mathbb{Z}_N \) such that \( \Phi^p l = 0 \). These distinct order \( p \) subgroups will be the subgroups we are trying to distinguish. By the reduction of Bacon et. al., distinguishing these is enough to solve the HSP (because, as they explain, we can identify the trivial subgroup in a constant number of queries).

It will be useful to consider the characters of both \( \mathbb{Z}_p \) and \( \mathbb{Z}_N \), so define

\[
\zeta := e^{2\pi i/p},
\]

\[
\omega := e^{2\pi i/N}.
\]

### 3.2 Overview of the algorithm

Our algorithm is presented in two parts: single-query and multi-query. We run the single-query algorithm in parallel many times. Each run may result in success or failure. The input to the multi-query algorithm is the tensor product of \( j \) copies of successful single-query runs.

We assume that \( CG \) is encoded into our quantum computer in a reasonable way so that we may write \( |(x, y)\rangle \) in two registers as \( |x\rangle |y\rangle \). With this assumption, it is easy to create the equal superposition state:

\[
(F_p \otimes F_N \otimes I) |0\rangle |0\rangle |0\rangle = \frac{1}{\sqrt{pN}} \sum_{j=0}^{p-1} \sum_{m=0}^{N-1} |j\rangle |m\rangle |0\rangle
\]

\[
= \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |0\rangle.
\]
Each single-query iteration follows the Standard Method. The Standard Method calls the oracle on an equal superposition of all group elements in the query register and zero in the response register:

$$O_{H_1} \left( \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |0\rangle \right) = \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |f_{H_1}(g)\rangle.$$ 

We then measure the response register to collapse to the state

$$|gH_1\rangle := \frac{1}{\sqrt{p}} \sum_{h \in H_1} |gh\rangle,$$

for a random $g \in G$. Suppose $g = (x,y)$. Then we have

$$|gH_1\rangle = \frac{1}{\sqrt{p}} \sum_{j=0}^{p-1} |x+j\rangle |k^jy + \Phi^j l\rangle.$$ 

By applying a series of quantum Fourier transforms and measurements to $|gH_1\rangle$, we are able, with a probability dependent only on $p$, to yield the state

$$|\psi\rangle = \frac{1}{\sqrt{pN}} \sum_{j=0}^{p-1} \sum_{m=0}^{N-1} \omega^{(kjy + \Phi^j l)m} |m\rangle.$$ 

We then use a partial measurement to collapse $|\psi^*\rangle$ into the space

$$V^\mu = \mathbb{C} |\mu\rangle + \mathbb{C} |−k\mu\rangle + \mathbb{C} |−\mu\rangle + \mathbb{C} |k\mu\rangle,$$

for some $\mu \in \mathbb{Z}_N$ and sum the amplitudes in the $|\mu\rangle$ and $|−k\mu\rangle$ components and the amplitudes in the $|−\mu\rangle$ and $|k\mu\rangle$ components to yield the state

$$|\hat{\psi}^\mu\rangle = \frac{1}{\sqrt{2}} (\omega^{\mu/2} |\mu\rangle + \omega^{-\mu/2} |−\mu\rangle).$$

This state is the same state Shakeel reaches at the end of one run of the single-query algorithm he presents for the dihedral group. Accordingly, we take the tensor product of many copies of $|\hat{\psi}^\mu\rangle$ for random values of $\mu$ and plug them into the multi-query algorithm, which proceeds exactly as in the dihedral case. With a subset sum solver we can identify $l$ with high probability with only $O(\log(N))$ queries.

Figures 3.1 and 3.2 on the next page present diagrams of quantum circuits implementing the single-query and multi-query algorithms respectively.
Chapter 3. The HSP for $\mathbb{Z}_N \times \mathbb{Z}_p$

Figure 3.1: The Single-Query Algorithm

Figure 3.2: The Multi-Query Algorithm
3.3 Preliminaries

As explained in the overview, in the single query-algorithm we need to project the vector $|\psi\rangle$ into the space $V^\mu$ for some $\mu \in \mathbb{Z}_N$. The spaces $V^\mu$ over all $\mu \in \mathbb{Z}_N$ clearly overlap. The aim of this section is to show how we can nevertheless project $|\psi^\ast\rangle$ into these subspaces, and in particular collapse $|\psi\rangle$ into $V^\mu$ with $\mu$ a random element of a large subset of $\mathbb{Z}_N$.

First we need the following result about the effect of multiplication by $k$ on an element of $\mathbb{Z}_N$:

**Claim 3.2.** For each $m \in \mathbb{Z}_N$, either $km = m$ or $k^jm \neq m$ for any $0 < j < N$.

**Proof:** Fix some $m \in \mathbb{Z}_N$ and let $j$ be the smallest positive integer such that $k^jm = m$. Then note $k^pm = m = k^j m$ and $k^j$ has a multiplicative inverse, so in fact $k^p-jm = m$. We can repeatedly multiply by $k^{-j}$ to arrive at $k^r m = m$ for $r < j$. But by the minimality of $j$, it follows that $r = 0$. Thus $j$ divides $p$, which means $j = 1$ or $j = p$. □

The $m \in \mathbb{Z}_N$ such that $km = m$ are in some sense badly behaved for our purposes; accordingly, define the set

$$\mathbb{Z}_N^1 := \{m \in \mathbb{Z}_N : km \neq m\}.$$ 

If we set $q := N/\gcd(N,k-1)$, then $\mathbb{Z}_N^1 = \{m \in \mathbb{Z}_N : m \neq qy \text{ for all } y \in \mathbb{Z}_N\}$. In particular, $q \geq 2$, so $|\mathbb{Z}_N| \geq \frac{N}{2}$. For $N$ prime, $\mathbb{Z}_N^1 = \mathbb{Z}_N \setminus \{0\}$.

For each $m \in \mathbb{Z}_N$, define the set

$$\mathcal{O}_m := \{m, km, k^2m, \ldots, k^{p-1}m\}.$$ 

Obviously the $\mathcal{O}_m$ partition $\mathbb{Z}_N$. By the above claim, for $m \in \mathbb{Z}_N^1$, we have $|\mathcal{O}_m| = p$ and for $m \in \mathbb{Z}_N \setminus \mathbb{Z}_N^1$, we have $|\mathcal{O}_m| = 1$. Note that $m \in \mathbb{Z}_N^1$ if and only if $-m \in \mathbb{Z}_N^1$, and that $\mathcal{O}_m$ and $\mathcal{O}_{-m}$ are disjoint unless $m$ is 0 or $N/2$, which are never elements of $\mathbb{Z}_N^1$. The spaces $V^\mu$ that we wish to project into are subspaces of $\mathbb{C}(\mathcal{O}_m \cup \mathcal{O}_{-m})$. We now explain how to associate one $V^\mu$ to each $\mathcal{O}_m \cup \mathcal{O}_{-m}$ by choosing a random representative of this set every single-query iteration.

For $i \geq j$ with $i, j \in \mathbb{Z}$, define the operator $\text{ord}_j \{a_1, \ldots, a_i\}$ to be the $j$th element of the set $\{a_1, \ldots, a_i\}$ when ordered from least to greatest. We will take $a_1, \ldots, a_i \in \mathbb{Z}_N$ with the order defined as $0 < 1 < \cdots < N-1$. For each iteration of the single-query algorithm, choose the number $r$ randomly from $\{1, \ldots, 2p\}$. We then define the set of representatives $\mathbf{R}_r$ as

$$\mathbf{R}_r := \{m \in \mathbb{Z}_N^1 : \text{ord}_r(\mathcal{O}_m \cup \mathcal{O}_{-m}) = m\}.$$ 

Checking membership in the set of $\mathbf{R}_r$ is classically efficient since it takes $\text{poly}(p)$ steps. Thus we can also efficiently compute the function $g : \mathbb{Z}_N \rightarrow \mathbb{Z}_N$, defined as

$$g(m) = \begin{cases} 
  m &: m \in \mathbf{R}_r \\
  -km &: -km \in \mathbf{R}_r \\
  -m &: -m \in \mathbf{R}_r \\
  km &: km \in \mathbf{R}_r \\
  m &: \text{otherwise}.
\end{cases}$$
Chapter 3. The HSP for $\mathbb{Z}_N \times \mathbb{Z}_p$

In the single-query algorithm, we will create the vector
\[
|\psi\rangle = \frac{1}{\sqrt{pN}} \sum_{j=0}^{p-1} \sum_{m=0}^{N-1} \omega^{(k^j y + \Phi^j t)m} |m\rangle.
\]
for a random $y \in \mathbb{Z}_N$. With an auxiliary qubit of $|0\rangle$, we will apply the transform $\alpha: C\mathbb{Z}_N \otimes C\mathbb{Z}_N \rightarrow C\mathbb{Z}_N \otimes C\mathbb{Z}_N$ defined as
\[
\alpha: |m\rangle |s\rangle \mapsto |m\rangle |g(m) + s\rangle,
\]
to $|\psi\rangle$ and then measure the second register to hopefully project this vector into one of the states
\[
V^{\mu} := C|\mu\rangle + C|-k\mu\rangle + C|-\mu\rangle + C|k\mu\rangle,
\]
for $\mu \in \mathbb{R}_r$.

Example 3.3. In the case of $N = 13, p = 3, k = 3, r = 1$, we have $\mathbb{R}_r = \{1, 2\}$ with
\[
V^1 = C|1\rangle + C|10\rangle + C|12\rangle + C|3\rangle,
\]
\[
V^2 = C|2\rangle + C|7\rangle + C|11\rangle + C|6\rangle.
\]
The left over spaces are $C|0\rangle, C|4\rangle, C|5\rangle, C|8\rangle$ and $C|9\rangle$.

Claim 3.4. For $m \in \mathbb{Z}_N$, define,
\[
\text{prob}_{\text{avg}}(m) := \frac{1}{pN^2} \sum_{y=0}^{N-1} \left| \sum_{j=0}^{p-1} \omega^{(k^j y + \Phi^j t)m} \right|^2.
\]

Note that $\text{prob}_{\text{avg}}$ is the average probability, over all values of $y$, of measuring the value $m$ if we were to measure $|\psi\rangle$. We claim that $\text{prob}_{\text{avg}}(m) = \frac{1}{N}$ for $m \in \mathbb{Z}_N^\dagger$.

Proof: We compute
\[
= \frac{1}{pN^2} \sum_{y=0}^{N-1} \left( \sum_{j=0}^{p-1} \omega^{(k^j y + \Phi^j t)m} \right) \left( \sum_{j=0}^{p-1} \omega^{-(k^j y + \Phi^j t)m} \right)
= \frac{1}{pN^2} \sum_{y=0}^{N-1} \left( p + 2 \sum_{i>j \in \mathbb{Z}_p} \text{Re} \left( \omega^{(k^i y + \Phi^i t)m} \omega^{-(k^j y + \Phi^j t)m} \right) \right)
= \frac{1}{N} \frac{2}{pN^2} \sum_{y=0}^{N-1} \sum_{i>j \in \mathbb{Z}_p} \text{Re} \left( \omega^{(k^i y + \Phi^i t-k^j y - \Phi^j t)m} \right).
\]
3.4 Single-query algorithm

Fix some \( i > j \in \mathbb{Z}_p \). We wish to show

\[
\sum_{y=0}^{N-1} \text{Re} \left( \omega^{(k^i y + \Phi^i l - k^j y - \Phi^j l)m} \right) = 0.
\]

Of course, it suffices to show

\[
\sum_{y=0}^{N-1} \omega^{(k^i y + \Phi^i l - k^j y - \Phi^j l)m} = 0.
\]

Then note

\[
\sum_{y=0}^{N-1} \omega^{(k^i y + \Phi^i l - k^j y - \Phi^j l)m} = \zeta \sum_{y=0}^{N-1} \omega^{(k^i - k^j)my},
\]

for some \( \zeta \in \mathbb{C} \). But since \( km \neq m \), we know \( (k^i - k^j)m \neq 0 \). So set \( r := \text{gcd}((k^i - k^j)m, N) \). Then \( \omega^{(k^i - k^j)m} \) is a primitive \( r \)th root of unity, and thus

\[
\sum_{y=0}^{N-1} \omega^{(k^i - k^j)my} = \frac{N}{r} \sum_{y=0}^{r-1} \omega^{(k^i - k^j)my} = 0,
\]

which finishes proof of the claim. \( \Box \)

With this claim established, it follows that the probability that measuring the second register projects \( \alpha \ket{\psi} \) into a particular \( V^\mu \) for \( \mu \in \mathbb{R}_r \) is \( \frac{4}{N} \). In order to compute how likely we are to project \( \alpha \ket{\psi} \) into any \( V^\mu \), we must consider the size of \( \mathbb{R}_r \). We established previously that \( |\mathbb{Z}_N^\dagger| \geq \frac{N}{2} \), and since \( |\mathcal{O}_m \cup \mathcal{O}_{-m}| = 2p \) for all \( m \in \mathbb{Z}_N^\dagger \), we have that \( |\mathbb{R}_r| \geq \lfloor \frac{N}{4p} \rfloor \). Thus the probability of projecting into some \( V^\mu \) is greater than or equal to \( \frac{4}{N} \cdot \lfloor \frac{N}{4p} \rfloor \approx \frac{1}{p} \). For \( N \) prime, this probability is actually \( 2^{(N-1)} \approx \frac{2}{p} \). We can then choose when we have successfully projected into a \( V^\mu \) since we know \( r \) and we know \( \mu \), so we can check if \( \mu \in \mathbb{R}_r \). If we do not succeed we throw away that run of the single-query algorithm. When we do succeed, we have that \( \mu \) is a random element of \( \mathcal{O}_m \cup \mathcal{O}_{-m} \) for a random \( m \in \mathbb{Z}_N^\dagger \). Thus, since the size of all these \( \mathcal{O}_m \) are the same and they partition \( \mathbb{Z}_N^\dagger \), we have in fact chosen \( \mu \) uniformly at random from \( \mathbb{Z}_N^\dagger \).

3.4 Single-query algorithm

The Standard Method gives

\[
\ket{gH_i} = \frac{1}{\sqrt{p}} \sum_{j=0}^{p-1} \ket{x+j} \ket{k^j y + \Phi^j l}.
\]

Apply \( \mathcal{F}_p \otimes I \):

\[
(\mathcal{F}_p \otimes I) \ket{gH_i} = \frac{1}{p \sqrt{N}} \sum_{j=0}^{p-1} \left( \sum_{s=0}^{p-1} \zeta^{(x+j)s} \ket{s} \right) \otimes \ket{k^j y + \Phi^j l}.
\]
Measure the first tensor factor in the computational basis, and call this result $\nu \in \mathbb{Z}_p$. The probability that we measure $\nu = 0$ is given by

$$\Pr(\nu = 0) = \frac{1}{p^2} \sum_{j=0}^{p-1} |\zeta^j|^2 = \frac{1}{p}.$$ 

Assume $\nu = 0$; the state post-measurement becomes

$$\frac{1}{\sqrt{p}} \sum_{j=0}^{p-1} \sum_{s=0}^{p-1} |k^3 y + \Phi^j l\rangle.$$ 

Apply $F_N$ and call the resulting vector $|\psi\rangle$; we have

$$|\psi\rangle = \frac{1}{\sqrt{pN}} \sum_{j=0}^{p-1} \sum_{m=0}^{N-1} \omega^{(k^3 y + \Phi^j l)m} |m\rangle.$$ 

Now, as explained in the previous section, we generate the random number $r$ and use an auxiliary qubit of $|0\rangle$ to apply the transform $\alpha$:

$$\alpha |\psi\rangle = \frac{1}{\sqrt{pN}} \sum_{j=0}^{p-1} \sum_{m=0}^{N-1} \omega^{(k^3 y + \Phi^j l)m} |m\rangle |g(m)\rangle.$$ 

We then measure the second register to hopefully project into the spaces $V^\mu$. Assume we measure $\mu \in \mathbb{R}_r$ and call the resulting vector $|\psi^\mu\rangle$. We have, for a moment ignoring the global normalization scalar,

$$|\psi^\mu\rangle = \sum_{j=0}^{p-1} \omega^{(k^3 y + \Phi^j l)\mu} |\mu\rangle + \sum_{j=0}^{p-1} \omega^{-(k^3 y + \Phi^j l)k\mu} |\mu\rangle \quad \text{(for } |k\mu\rangle)$$

$$+ \sum_{j=0}^{p-1} \omega^{-(k^3 y + \Phi^j l)\mu} |\mu\rangle + \sum_{j=0}^{p-1} \omega^{(k^3 y + \Phi^j l)k\mu} |\mu\rangle.$$ 

We will now use an auxiliary qubit, which will be $|0\rangle$ for the $|\mu\rangle$ and $|\mu\rangle$ factors, and $|1\rangle$ for the $|-k\mu\rangle$ and $|k\mu\rangle$ factors, i.e. we have

$$|\psi^\mu\rangle = \sum_{j=0}^{p-1} \omega^{(k^3 y + \Phi^j l)\mu} |\mu\rangle |0\rangle + \sum_{j=0}^{p-1} \omega^{-(k^3 y + \Phi^j l)k\mu} |\mu\rangle |1\rangle$$

$$+ \sum_{j=0}^{p-1} \omega^{-(k^3 y + \Phi^j l)\mu} |\mu\rangle |0\rangle + \sum_{j=0}^{p-1} \omega^{(k^3 y + \Phi^j l)k\mu} |\mu\rangle |1\rangle.$$ 

We now perform a controlled permutation on the first register that sends $|-k\mu\rangle$ to $|\mu\rangle$ and $|k\mu\rangle$ to $|\mu\rangle$ if the second register is $|1\rangle$ and acts as the identity if the
second register is $|0\rangle$. We then we apply a Hadamard transform on the second register. Calling the resulting vector $|\tilde{\psi}^\mu\rangle$, we have

$$
|\tilde{\psi}^\mu\rangle = \sum_{j=0}^{p-1} \omega^{(k^j y + \Phi^j l)\mu} |\mu\rangle + \sum_{j=0}^{p-1} \omega^{-(k^j y + \Phi^j l)k\mu} |\mu\rangle -
$$

$$
+ \sum_{j=0}^{p-1} \omega^{-(k^j y + \Phi^j l)\mu} |\bar{\mu}\rangle + \sum_{j=0}^{p-1} \omega^{(k^j y + \Phi^j l)k\mu} |-\mu\rangle -
$$

Measure the second register in the computational basis. Suppose we measure 0. Then, calling the result $|\hat{\psi}^\mu\rangle$, we have

$$
|\hat{\psi}^\mu\rangle = \sum_{j=0}^{p-1} \omega^{(k^j y + \Phi^j l)\mu} + \omega^{-(k^j y + \Phi^j l)k\mu} |\mu\rangle
$$

$$
+ \sum_{j=0}^{p-1} \omega^{-(k^j y + \Phi^j l)\mu} + \omega^{(k^j y + \Phi^j l)k\mu} |-\mu\rangle
$$

Clearly these two amplitudes are conjugates of one another and thus the components have the same magnitude. But also note that we have

$$
\sum_{j=0}^{p-1} \omega^{(k^j y + \Phi^j l)\mu} + \omega^{-(k^j y + \Phi^j l)k\mu} = c\omega^{l\mu/2},
$$

for some $c \in \mathbb{R}$. To see this, recall that $\Phi^j l = 0$ and perform the following computation:

$$
\omega^{3\mu} + \omega^{(ky+l)\mu} + \ldots + \omega^{(k^{p-1}y+(k^{p-2}+\ldots+1)l)\mu}
$$

$$
+ \omega^{-ky\mu} + \omega^{-k(ky+l)\mu} + \ldots + \omega^{-k(k^{p-1}y+(k^{p-2}+\ldots+1)l)\mu}
$$

$$
= \omega^{l\mu}(\omega^{(y+k^{p-1}l+\ldots+kl)\mu} + \omega^{ky\mu} + \ldots + \omega^{(k^{p-1}y+k^{p-2}l+\ldots+kl)\mu})
$$

$$
+ (\omega^{-ky\mu} + \omega^{(-k^2y-kl)\mu} + \ldots + \omega^{(-y-k^{p-1}l-\ldots-kl)\mu})
$$

$$
= \omega^{l\mu} z + \bar{z}
$$

for some $z \neq 0 \in \mathbb{C}$. (We have $z \neq 0$ because otherwise we would not have projected into $V^\mu$.) But then observe that

$$
\omega^{l\mu} z + \bar{z} = \omega^{l\mu/2}(\omega^{l\mu/2} z + \omega^{-l\mu/2} \bar{z})
$$

$$
= \omega^{l\mu/2}(\zeta + \bar{\zeta})
$$

for some $\zeta \neq 0 \in \mathbb{C}$, and of course $(\zeta + \bar{\zeta}) \in \mathbb{R}$. So the state in fact projects to

$$
|\hat{\psi}^\mu\rangle = \frac{1}{\sqrt{2}} \left( \omega^{l\mu/2} |\mu\rangle + \omega^{-l\mu/2} |-\mu\rangle \right).
$$
Note that \((\zeta + \overline{\zeta}) \neq 0\), because otherwise we would not have measured 0 in the auxiliary register. So suppose instead we measure 1. We have

\[
|\hat{\psi}^\mu\rangle = \sum_{j=0}^{p-1} \omega^{(k^j y + \Phi^j l)\mu} - \omega^{-(k^j y + \Phi^j l)k\mu} |\mu\rangle
\]

\[+ \sum_{j=0}^{p-1} \omega^{-(k^j y + \Phi^j l)\mu} - \omega^{(k^j y + \Phi^j l)k\mu} |-\mu\rangle.
\]

Substantially similar calculations to the previous case show that

\[
|\hat{\psi}^\mu\rangle = \frac{i}{\sqrt{2}} \left(\omega^{l\mu/2} |\mu\rangle + \omega^{-l\mu/2} |-\mu\rangle\right).
\]

But we can correct for the global phase by applying the transform \(|m\rangle \mapsto -i |m\rangle\), so in fact we have

\[
|\hat{\psi}^\mu\rangle = \frac{1}{\sqrt{2}} \left(\omega^{l\mu/2} |\mu\rangle + \omega^{-l\mu/2} |-\mu\rangle\right).
\]

### 3.5 Multi-query algorithm

To reach the state \(\frac{1}{\sqrt{2}} \left(\omega^{l\mu/2} |\mu\rangle + \omega^{-l\mu/2} |-\mu\rangle\right)\), we need \(\nu = 0\) and \(\mu \in \mathbb{R}_r\). Call such a run of the single-query algorithm a **successful run**. Our input into the multi-query algorithm will be the tensor product of \(j\) successful runs; that is, we start with the tensor product of \(j\) samples of these \(|\hat{\psi}^\mu\rangle\)'s. We will use the vector \(\vec{\mu} := (\mu_1, \ldots, \mu_j) \in (\mathbb{Z}_N^j)^j\) to index the \(\mu\)'s. So our state at the beginning of the multi-query algorithm is

\[
|\psi^{\vec{\mu}}\rangle := \bigotimes_{m=1}^{j} |\hat{\psi}^{\mu_m}\rangle,
\]

for some \(\vec{\mu}\) chosen uniformly at random among elements of \((\mathbb{Z}_N^j)^j\).

We now introduce a version the subset sum problem and demonstrate its connection to our HSP algorithm.

**Definition 3.5.** Let \(\vec{b} = (b_1, \ldots, b_k)\) be an element of \((-1, 1)^j\) and define

\[
\vec{b} \cdot \vec{\mu} := \sum_{m=1}^{j} b_m \mu_m \pmod{2N}.
\]

For any \(z \in \mathbb{Z}_{2N}\), we define

\[
\hat{S}_{\xi}^{\vec{\mu}} := \{\vec{b} \in \{-1, 1\}^j : \vec{b} \cdot \vec{\mu} = z\}.
\]

The **signed subset sum problem** is, given some \(\vec{\mu}\) and \(z\), find the set \(\hat{S}_{\xi}^{\vec{\mu}}\).
With this subset sum notation, we can write our vector $|\psi^\mu\rangle$ in a more sophisticated way:

$$|\psi^\mu\rangle = \frac{1}{\sqrt{2}} (\omega^{\mu_1 l/2} |\mu_1\rangle + \omega^{-\mu_1 l/2} |\mu_1\rangle) \otimes \cdots \otimes \frac{1}{\sqrt{2}} (\omega^{\mu_j l/2} |\mu_j\rangle + \omega^{-\mu_j l/2} |\mu_j\rangle)$$

$$= \sum_{\vec{b} \in \{-1, 1\}^j} \frac{1}{\sqrt{2}^j} \omega^{\vec{b} \cdot \vec{\mu}/2} \bigotimes_{m=1}^j |b_m \mu_m\rangle$$

$$= \frac{1}{\sqrt{2}^j} \sum_{z \in \mathbb{Z}_2N} \omega^{zl/2} \sum_{\vec{b} \in \hat{S}_z^\mu} \bigotimes_{m=1}^j |b_m \mu_m\rangle .$$

Since the vector $\vec{\mu}$ is known to us, it makes sense to identify $\bigotimes_{m=1}^j |b_m \mu_m\rangle$ with $|\vec{b}\rangle$. Under this identification we have

$$|\psi^\mu\rangle = \frac{1}{\sqrt{2}^j} \sum_{z \in \mathbb{Z}_2N} \omega^{zl/2} \sum_{\vec{b} \in \hat{S}_z^\mu} |\vec{b}\rangle .$$

Now we will suppose we have a solution to the signed subset sum problem. The sets $\hat{S}_z^\mu$ form a partition of all the $\vec{b} \in \{-1, 1\}^j$, so for any $z \neq z' \in \mathbb{Z}_2N$, the vectors $\sum_{\vec{b} \in \hat{S}_z^\mu} |\vec{b}\rangle$ and $\sum_{\vec{b} \in \hat{S}_{z'}^\mu} |\vec{b}\rangle$ are orthogonal. Define $\hat{n}_z^\mu := |\hat{S}_z^\mu|$ and define

$$Z^\vec{\mu} := \{ z \in \mathbb{Z}_2N : \hat{n}_z^\mu \neq 0 \}.$$ 

Then the vectors $(1/\sqrt{\hat{n}_z^\mu}) \sum_{\vec{b} \in \hat{S}_z^\mu} |\vec{b}\rangle$ for $z \in Z^\vec{\mu}$ are orthonormal, so we can find some unitary transformation $\beta$ that sends $(1/\sqrt{\hat{n}_z^\mu}) \sum_{\vec{b} \in \hat{S}_z^\mu} |\vec{b}\rangle \mapsto |z\rangle$. In order to apply the transform $\beta$ we need to be able to quantum sample subset sum solutions; that is, we need to be able to create an equal superposition over all vectors $\vec{b}$ such that $\vec{b} \cdot \vec{\mu} = z$ for any $z \in \mathbb{Z}_2N$. Applying $\beta$, we have

$$\beta |\psi^\mu\rangle = \frac{1}{\sqrt{2}^j} \sum_{z \in Z^\vec{\mu}} \sqrt{\hat{n}_z^\mu} \omega^{zl/2} |z\rangle .$$

We apply the inverse quantum Fourier transform on $\mathbb{Z}_2N$:

$$\mathcal{F}^{-1}_{2N} \beta |\psi^\mu\rangle = \frac{1}{\sqrt{2^j \sqrt{2N}}} \sum_{y \in \mathbb{Z}_{2^N}} \left( \sum_{z \in Z^\vec{\mu}} \sqrt{\hat{n}_z^\mu} \omega^{zl/2} \omega^{-y/2} \right) |y\rangle$$

$$= \frac{1}{\sqrt{2^{j+1} \mu}} \sum_{y \in \mathbb{Z}_{2N}} \left( \sum_{z \in Z^\vec{\mu}} \sqrt{\hat{n}_z^\mu} \omega^{z(l-y)/2} \right) |y\rangle .$$

Measure in the computational basis. Note that when we plug in $y = l$, we get $\omega^{z(l-y)/2} = 1$. When we plug in $y = l + N$, we get $\omega^{z(l-y)/2} = e^{-iz\pi}$, which is 1 if $z$ is
even and \(-1\) if \(z\) is odd. Our success probability is the probability of measuring \(l\) or \(l + N\) (from which we can obtain \(l\) since we know \(N\)):

\[
p_{\text{succ}, \vec{\mu}} = \frac{1}{2^{j+1}N} \left( \sum_{z \in \mathbb{Z}_{2N}} \sqrt{n_z^\mu} \right)^2 + \frac{1}{2^{j+1}N} \left( \sum_{z \in \mathbb{Z}_N} \sqrt{n_{2z}^\mu} - \sum_{z \in \mathbb{Z}_N} \sqrt{n_{2z+1}^\mu} \right)^2
\]

Since \(2N\) is even, the elements of \(\mathbb{Z}^\mu\) are either all even or all odd. Thus we have

\[
p_{\text{succ}, \vec{\mu}} = \frac{1}{2^{j+1}N} \left( \sum_{z \in \mathbb{Z}_{2N}} \sqrt{n_z^\mu} \right)^2.
\]

Our \(\vec{\mu}\) is chosen uniformly at random from \((\mathbb{Z}_N^\dagger)^j\). So our overall probability of success is

\[
p_{\text{succ}} = \frac{1}{2^j N |\mathbb{Z}_N^\dagger|^j} \sum_{\vec{\mu} \in (\mathbb{Z}_N^\dagger)^j} \left( \sum_{z \in \mathbb{Z}_{2N}} \sqrt{n_z^\mu} \right)^2.
\]

### 3.6 Analysis of probability of success

In analyzing this probability of success, it will help to consider another, more standard version of the subset sum problem which is in fact equivalent.

**Definition 3.6.** Let \(\vec{d} = (d_1, \ldots, d_k)\) be an element of \(\{0, 1\}^j\) and define

\[
\vec{d} \cdot \vec{\mu} := \sum_{m=1}^j d_m \mu_m \pmod{N}.
\]

For any \(z \in \mathbb{Z}_N\), we define

\[
S^\vec{\mu}_z := \{\vec{d} \in \{0, 1\}^j : \vec{d} \cdot \vec{\mu} = z\}.
\]

The *standard subset sum problem* is, given some \(\vec{\mu}\) and \(z\), find the set \(S^\vec{\mu}_z\).

As stated earlier, the elements of \(\mathbb{Z}^\vec{\mu}\) are either all even or all odd. Consider only the \(w \in \mathbb{Z}_{2N}\) of the same parity as elements in \(\mathbb{Z}^\vec{\mu}\). Then there is a bijection between the sets \(\hat{C}^\vec{\mu}_w\) and the sets \(S^\vec{\mu}_z\) for \(z \in \mathbb{Z}_N\) that preserves cardinalities. For any \(\vec{\beta} \in \{-1, 1\}^j\), let \(\vec{d}\) be given by \(d_i = 0 \iff b_i = -1\) and \(d_i = 1 \iff b_i = 1\). Note that \(\vec{\beta} \cdot \vec{\mu} = (-1, \ldots, -1) \cdot \vec{\mu} + 2(\vec{d} \cdot \vec{\mu})\). So if we let \(z := \vec{d} \cdot \vec{\mu}\) and \(c := (-1, \ldots, -1) \cdot \vec{\mu}\), we have \(|\hat{C}^\vec{\mu}_{2z+c}| = |S^\vec{\mu}_z|\).

Thus, defining \(n^\vec{\mu}_z := |S^\vec{\mu}_z|\), we can write the probability of success of the multi-query algorithm as

\[
p_{\text{succ}} = \frac{1}{2^j N |\mathbb{Z}_N^\dagger|^j} \sum_{\vec{\mu} \in (\mathbb{Z}_N^\dagger)^j} \left( \sum_{z \in \mathbb{Z}_N} \sqrt{n^\vec{\mu}_z} \right)^2.
\]

Our probability of success is very similar to that of Bacon et al. [4] However, whereas their \(\vec{\mu}\) is chosen uniformly at random from \(\mathbb{Z}_N^j\), ours is chosen from \((\mathbb{Z}_N^\dagger)^j\). We must show that this does not matter and that the \(\vec{d}\)'s spread out evenly among the \(S^\vec{\mu}_z\) for \(j\) on the order of \(\log(N)\).
3.6. Analysis of probability of success

3.6.1 A lemma on the distribution of \( d^j \)s

**Lemma 3.7.** For a fixed \( z \in \mathbb{Z}_N \) and \( \bar{\mu} \) uniformly random from \((\mathbb{Z}_N \setminus \{0\})^j\), define

\[
\rho(j) := \Pr \left( n_z^\bar{\mu} \geq \frac{3}{4} \left( \frac{2^j}{N} \right) \right).
\]

Then we have \( \rho(j) \geq 0.99 \) for \( j \geq 2 \log(N) + 10 \).

We first recap some work done in Regev [16]. Fix some \( z \in \mathbb{Z}_N \), and consider a uniformly random \( \vec{\tau} \in \mathbb{Z}_N^j \). For each \( \vec{d} \in \{0,1\}^j \), with \( \vec{d} \neq 0^j \), we define the random variable \( X_{\vec{d}} \) to be 1 if \( \vec{d} \cdot \vec{\tau} = z \) and 0 otherwise. Because the sum \( \sum_i d_i u_i \) (mod \( N \)) is every value with equal probability, we have \( E [X_{\vec{d}}] = \frac{1}{N} \). Thus \( E[n_z^\bar{\mu}] \geq E \left[ \sum X_{\vec{d}} \right] \leq N \). Further, Regev shows that for any \( \vec{d} \neq \vec{d'} \), the variables \( X_{\vec{d}} \) and \( X_{\vec{d'}} \) are independent. Thus we can apply the Chernoff bound to give

\[
\Pr \left( \sum_{\vec{d} \in \{0,1\}^j} X_{\vec{d}} < \left( \frac{3}{4} \right) \frac{2^j}{N} \right) \leq e^{-\frac{(2^j-1)^2}{32N}}.
\]

Note that \( n_z^\bar{\mu} \) is either \( \sum X_{\vec{d}} \) or \( \sum X_{\vec{d}} + 1 \), so \( \sum X_{\vec{d}} < \left( \frac{3}{4} \right) \frac{2^j}{N} \) implies that \( n_z^\bar{\mu} < \left( \frac{3}{4} \right) \frac{2^j}{N} \). So let us define \( \sigma(j) := \Pr \left( n_z^\bar{\mu} \geq \left( \frac{3}{4} \right) \frac{2^j}{N} \right) \). Then we have

\[
\sigma(j) \geq 1 - e^{-\frac{(2^j-1)^2}{32N}}. \tag{3.2}
\]

**Claim 3.8.** A little combinatorial work shows that

\[
\sigma(j) = \sum_{i=0}^{j} \binom{j}{i} \frac{(N-1)^i}{N^j} \rho(i). \tag{3.3}
\]

**Proof:** For \( i = 0, \ldots, j \), let \( \vec{\tau}_i \) be a random element of \( \mathbb{Z}_N^j \), that has exactly \( i \) entries not equal to 0. The probability that \( n_z^\vec{\tau}_i \) is greater than \( \frac{3}{4} \left( \frac{2^j}{N} \right) \) is the sum of the probabilities of that same event for each of the \( \vec{\tau}_i \), weighted by their relative proportion; i.e., we have

\[
\sigma(j) = \sum_{i=0}^{j} \binom{j}{i} \frac{(N-1)^i}{N^j} \Pr \left( n_z^\vec{\mu} \geq \frac{3}{4} \left( \frac{2^j}{N} \right) \right).
\]

Let \( \vec{\mu}_i \) be a random element of \((\mathbb{Z}_N \setminus \{0\})^i\). Without loss of generality, we may assume \( \vec{\tau}_i = \vec{\mu}_i + 0^{j-i} \), where addition denotes tuple concatenation. Then consider any \( \vec{d} \) in \( n_z^\vec{\mu}_i \). For any \( \vec{d'} \in \{0,1\}^{j-i} \), we have that \( (\vec{d} + \vec{d'}) \) is an element of \( n_z^{\vec{\tau}_i} \). Conversely, any \( \vec{d} \) in \( n_z^{\vec{\tau}_i} \) corresponds to some \( \vec{d'} \) in \( n_z^{\vec{\mu}_i} \) if we drop its last \((j-i)\) components. Thus \( n_z^{\vec{\tau}_i} = 2^{j-i} n_z^{\vec{\mu}_i} \). So \( \Pr \left( n_z^{\vec{\tau}_i} \geq \frac{3}{4} \left( \frac{2^j}{N} \right) \right) = \Pr \left( n_z^{\vec{\mu}_i} \geq \frac{3}{4} \left( \frac{2^j}{N} \right) \right) = \rho(i) \), establishing our claim. \( \square \)
Of course $0 \leq \rho(i) \leq 1$ for all $i = 0, \ldots, j - 1$, so (3.3) gives us

$$\rho(j) \geq 1 - (1 - \sigma(j)) \left( \frac{N}{N - 1} \right)^j.$$ 

Using (3.2) and the fact that $\frac{N}{N - 1} \leq 2$, this becomes

$$\rho(j) \geq 1 - \frac{2^j}{e^{\frac{2^j}{e^{N-1}}}}.$$ 

If $j \geq 2 \lg(N) + 10$ then

$$\rho(j) \geq 1 - \frac{2^{10}N^2}{e^{16e^N}},$$

and since $\frac{N^2}{e^N} \leq 1$ for all $N \geq 2$, we have our result.

### 3.6.2 $N$ prime

In the case of $N$ prime, $\mathbb{Z}_N^+ = \mathbb{Z}_N \setminus \{0\}$. Then by Cauchy’s inequality applied to (3.1), we have

$$p_{\text{succ}} \geq \frac{1}{2^j N} \left( \frac{1}{(N-1)^j} \sum_{\overrightarrow{m} \in (\mathbb{Z}_N \setminus \{0\})^j} \sum_{z \in \mathbb{Z}_N} \sqrt{n_z^\overrightarrow{m}} \right)^2.$$ 

But by Lemma 3.7, for any $z \in \mathbb{Z}_N$, with $j \geq 2 \lg(N) + 10$,

$$\frac{1}{(N-1)^j} \sum_{\overrightarrow{m} \in (\mathbb{Z}_N \setminus \{0\})^j} \sqrt{n_z^\overrightarrow{m}} \geq \sqrt{\frac{3}{4} \left( \frac{2^j}{N} \right)^2} \Pr \left( n_z^\overrightarrow{m} \geq \frac{3}{4} \left( \frac{2^j}{N} \right) \right) \geq (0.99) \sqrt{\frac{3}{4} \left( \frac{2^j}{N} \right)}.$$ 

So, combining (3.4) and (3.5), we have

$$p_{\text{succ}} \geq \frac{1}{2^j N} \left( \sum_{z \in \mathbb{Z}_N} (0.99) \sqrt{\frac{3}{4} \left( \frac{2^j}{N} \right)} \right)^2 = (0.99)^2 \frac{3}{4} > \frac{2}{3}.$$ 

### 3.6.3 General $N$

Recall that $\mathbb{Z}_N^+ = \{ z \in \mathbb{Z}_N : z \neq qy \text{ for all } y \in \mathbb{Z}_N \}$. Fix some $z \in \mathbb{Z}_N$ and let $\overrightarrow{m} \in (\mathbb{Z}_N^+)^j$ be uniformly random. We can represent each component of $\overrightarrow{m}$ as $\mu_i = \alpha_i q + \beta_i$, where the $\alpha_i$ is a random element of $\{0, \ldots, N/q\}$ and the $\beta_i$ is a random element of $\{1, \ldots, q - 1\}$. Then $\overrightarrow{d} \cdot \overrightarrow{m} = \sum_i d_i \mu_i \pmod{N} = \gamma q + (\sum_i d_i \beta_i \pmod{q})$, where $\gamma$ is a random element of $\{0, \ldots, N/q\}$.

If we let $\overrightarrow{n}_z^\overrightarrow{m} = \{ \overrightarrow{d} \in \{0, 1\}^j : \overrightarrow{d} \cdot \overrightarrow{m} \pmod{q} = z \pmod{q} \}$, then Lemma 1 gives us

$$\Pr \left( \overrightarrow{n}_z^\overrightarrow{m} \geq \frac{3}{4} \left( \frac{2^j}{q} \right) \right) \geq 0.99,$$

(3.6)
3.7. Conclusion

for \( j \geq 2 \lg(q) + 10 \).

But then note that the event \( \vec{d} \cdot \vec{\mu} \pmod{q} = z \pmod{q} \) and the event \( [(\vec{d} \cdot \vec{\mu})/q] = [z/q] \) are independent, because \( [(\vec{d} \cdot \vec{\mu})/q] \) takes on every value in \( \{0, \ldots, N/q\} \) with equal probability regardless of what the \( \beta_i \) are.

For each \( \vec{d} \in \tilde{n}_{\vec{\mu}}^z \), let \( X_{\vec{d}} = 1 \) if \( \lfloor (\vec{d} \cdot \vec{\mu})/q \rfloor = \lfloor z/q \rfloor \) and \( X_{\vec{d}} = 0 \) otherwise. Again, \( \lfloor (\vec{d} \cdot \vec{\mu})/q \rfloor \) takes on every value in \( \{0, \ldots, N/q\} \) with equal probability. So

\[
\Pr \left( n_{\vec{\mu}}^z \geq \frac{5}{6} \left( \frac{q \tilde{n}_{\vec{\mu}}^z}{N} \right) \right) \geq 1 - e^{-\frac{q \tilde{n}_{\vec{\mu}}^z}{27N}}. \tag{3.7}
\]

So, combining (3.6) and (3.7), we have

\[
\Pr \left( n_{\vec{\mu}}^z \geq \frac{5}{8} \left( \frac{2^j}{N} \right) \right) \geq \Pr \left( n_{\vec{\mu}}^z \geq \frac{5}{6} \left( \frac{q \tilde{n}_{\vec{\mu}}^z}{N} \right) \mid \tilde{n}_{\vec{\mu}}^z \geq \frac{3}{4} \left( \frac{2^j}{q} \right) \right) \Pr \left( \tilde{n}_{\vec{\mu}}^z \geq \frac{3}{4} \left( \frac{2^j}{q} \right) \right)
\]

\[
\geq \left( 1 - e^{-\frac{2^j}{96N}} \right) (0.99) > (0.99)^2 > 0.98,
\]

for \( j \geq 2 \lg(N) + 10 \). Then the same calculations as in the case of \( N \) prime show that

\[
p_{\text{succ}} \geq (0.98)^2 \frac{5}{8} > 0.6.
\]

3.7 Conclusion

To successfully identify the hidden subgroup \( H_l \) with probability greater than some fixed constant greater than \( 1/2 \), we need \( O(\log(N)) \) successful runs. We measure \( \nu = 0 \) with probability \( 1/p \) and we measure \( \mu \in \mathbb{R}_r \) with probability at least \( 1/p \), so we need only \( O(p^2 \log(N)) \) queries to yield \( O(\log(N)) \) successful runs. But since \( N/p \neq \text{poly}(\log(N)) \), this is just \( O(\log(N)) \) queries. Thus we have presented an efficient algorithm for solving the HSP on \( \mathbb{Z}_N \rtimes_k \mathbb{Z}_p \) under the assumption that we can efficiently quantum sample solutions to the subset sum problem. We remark that, as in the case of the dihedral HSP, the subset sum solver is required only for the multi-query algorithm and that the single-query algorithm may be independently of interest.
Chapter 4

Representation Theory and the HSP for $\mathbb{Z}_N \rtimes \mathbb{Z}_p$

In the previous chapter we presented an efficient algorithm for solving the HSP for $\mathbb{Z}_N \rtimes \mathbb{Z}_p$ under the assumption that there is an efficient implementation of the Pretty Good Measurement for the dihedral HSP. Our approach was to mimic the algorithm of Bacon et al. [4] for the dihedral group wherever possible; however, we provided no justification for many steps of our procedure. In this chapter, we motivate some of our decisions from the perspective of representation theory. In particular, we give some reasons for why it is appropriate to apply a series of Fourier transforms on each component and why we would want to project our state into a subspace of $\mathbb{C} \mathbb{Z}_N^\dagger$. For a quick overview of the representation theory of finite groups used in this section, see Appendix A.

4.1 The Fourier transform over a group

Earlier we considered the discrete Fourier transform, but it is in fact possible to define the Fourier transform over any arbitrary finite group. We shall see that the Fourier transform over the cyclic group $\mathbb{Z}_N$ is exactly the discrete Fourier transform defined in chapter 1. Let $G$ be a finite group and let $\hat{G}$ be a complete set of inequivalent irreducible representations of $G$ (see Fact A.5 for the definition of a complete set of irreps). Note that the Fourier transform over a group depends, as the definition below makes clear, on the set $\hat{G}$ of irreducible representations we choose. Define the Fourier coefficients $[\rho(g)]_{a,b}$, for $g \in G$ and $\rho \in \hat{G}$ of dimension $d_\rho$ with $1 \leq a, b \leq d_\rho$, to bet the $(a,b)$th entry of the matrix $\rho(g)$.

**Definition 4.1.** The quantum Fourier transform over a finite group $G$, $\mathcal{F}_G$, acts on a basis of $\mathbb{C}G$ by

$$\mathcal{F}_G |g\rangle = \sum_{\rho \in \hat{G}} \sum_{1 \leq a, b \leq d_\rho} \sqrt{\frac{d_\rho}{|G|}} [\rho(g)]_{a,b} |\rho, a, b\rangle.$$
Chapter 4. Representation Theory and the HSP for \( \mathbb{Z}_N \times \mathbb{Z}_p \)

The sum of the squares of the dimensions of the irreps equals the size of the group (see Fact A.5), so it is easy to see that there are the same number of basis vectors \(|\rho, a, b\rangle\) as \(|g\rangle\). For an overview of the quantum Fourier transform over a group, and its applications to the HSP, see the master’s thesis of Jean-Noël Murphy [14]. Murphy shows that \(\mathcal{F}_G\) is indeed a unitary transform, and thus in principle could be implemented by a quantum computer. However, there is no guarantee that there is an \textit{efficient} way of implementing the Fourier transform for an arbitrary group \(G\); that is, there is no known general decomposition of \(\mathcal{F}_G\) into a product of only a few simple transforms. Efficient implementations of the quantum Fourier transform do exist for many families of groups; see [17] for the cases of Abelian groups, the dihedral group, and the symmetric group.

All known efficient quantum solutions to the HSP for a group \(G\) are essentially based on the quantum Fourier transform over \(G\). To see where the Fourier transform comes into play, we redo some computations in Childs and van Dam [2], sections VI A and VII C.

\textbf{Definition 4.2.} The right regular representation \(R: G \rightarrow GL_{|G|}(\mathbb{C})\) acts on a basis of \(C^G\) by:

\[ R(g_1) |g_2\rangle = |g_2g_1^{-1}\rangle. \]

\textbf{Theorem 4.3.} Recall that \(|gH\rangle\) is the coset state that results from the Standard Method applied to the HSP. Define,

\[ \sigma_H := \frac{1}{|G|} \sum_{g \in G} |gH\rangle \langle gH| \]

to be the statistical ensemble (or mixed state) over all \(|gH\rangle\) for all possible \(g \in G\). Then the Fourier transform over \(G\) block-diagonalizes \(\sigma_H\), from which we sample as part of the Standard Method.

\textit{Proof:} We will show first that the Fourier transform \(\mathcal{F}_G\) breaks \(R\) into its irreducible components. If we write \(R(g_1) = \sum_{g_2 \in G} \sum_{\rho, \rho' \in \hat{G}} \frac{1}{d_{\rho, \rho'}} |\rho(g_2g_1^{-1})\rangle_{a, b} |\rho'(g_2)\rangle_{a', b'} \langle \rho, a, b| \langle \rho', a', b'|\)

\[ \hat{R}(g_1) = \sum_{g_2 \in G} \mathcal{F}_G |g_2g_1^{-1}\rangle \langle g_2| \mathcal{F}_G^\dagger, \]

\[ = \sum_{g_2 \in G} \mathcal{F}_G |g_2g_1^{-1}\rangle \langle g_2| \mathcal{F}_G^\dagger, \]

\[ = \sum_{g_2 \in G} \sum_{\rho, \rho' \in \hat{G}} \sum_{a, b, a', b'} d_{\rho, \rho'} \langle \rho'(g_2)\rangle_{a', b'} \langle \rho(g_2g_1^{-1})\rangle_{a, b} \langle \rho', a', b' | \rho, a, b \rangle \langle \rho', a', b' | \]

\[ = \sum_{\rho \in \hat{G}} \sum_{a, b, c=1} d_{\rho} \sum_{a', b'} \langle \rho(g_1)\rangle_{a, c} \langle \rho(g_1^{-1})\rangle_{c, b} \langle \rho'(g_2)\rangle_{a', b'} | \rho, a, b \rangle \langle \rho', a', b' | \]

\[ = \bigoplus_{\rho \in \hat{G}} (I_{d_{\rho}} \otimes \rho(g_1)), \]
where $I_{d_p}$ is the $d_p \times d_p$ identity matrix, and we have used the orthogonality of irreducible representations (see Lemma A.6) in the third line. We can write the coset state $|gH\rangle$ that results from the Standard Method as $|gH\rangle = \frac{1}{|H|} \sum_{h \in H} R(h) |g\rangle$. Then the mixed state $\sigma_H$ is

$$\sigma_H = \frac{1}{|G| \cdot |H|} \sum_{g \in G} \sum_{h_1, h_2 \in H} R(h_1) |g\rangle \langle g| R^\dagger(h_2)$$

$$= \frac{1}{|G| \cdot |H|} \sum_{h_1, h_2 \in H} R(h_1 h_2^{-1})$$

$$= \frac{1}{|G|} \sum_{h \in H} R(h).$$

The upshot of this computation is that, since the Fourier transform over $G$ block diagonalizes $R$, it is also the case that $\mathcal{F}_G$ block diagonalizes $\sigma_H$. □

We now will demonstrate that our HSP algorithm employs the Fourier transform over $\mathbb{Z}_N \rtimes_k \mathbb{Z}_p$, although we have previously not described it in those terms. Therefore we must investigate representations of $\mathbb{Z}_N \rtimes_k \mathbb{Z}_p$ and work out explicitly how the Fourier transform over $\mathbb{Z}_N \rtimes_k \mathbb{Z}_p$ acts.

### 4.2 Irreducible representations of $\mathbb{Z}_N \rtimes_k \mathbb{Z}_p$

We now compute the irreducible representations of $G := \mathbb{Z}_N \rtimes_k \mathbb{Z}_p$. We follow Serre [20], section 8.2, which explains how to use Mackey’s criterion to construct the irreducible representations of a semidirect product by an abelian group from representations of the component groups. See Serre for a precise statement of Mackey’s criterion.

Let $X = \text{Hom}(\mathbb{Z}_N, C^*)$ be the group of irreducible representations of $\mathbb{Z}_N$. The elements of $X$ are $\chi_m$ for $0 \leq m \leq N - 1$, where $\chi_m$ is determined by $\chi_m(1) = \omega^m$ for some primitive $N$th root of unity $\omega$. The group $G$ acts on $X$ by $$(g\chi)(y) = \chi(g^{-1}yg) \quad \text{for } g \in G, \chi \in X, y \in \mathbb{Z}_N.$$ We are interested in the orbits of $X$ under $\mathbb{Z}_p$ (considered as a subgroup of $G$). Fix some $\chi_m \in X$. Then

$$(x, 0) \cdot \chi_m(0, 1) = \chi_m((-x, 0) \cdot (0, 1) \cdot (x, 0)) = \chi_m(0, k^x) = \omega^{mk^x}.$$ Thus we see that $((x, 0) \cdot \chi_m) = \chi_{mk^x}$. There are two cases to consider. If $km = m$, then the orbit of $\chi_m$ under $\mathbb{Z}_p$ is just $\{\chi_m\}$. If $km \neq m$, then, as we have seen earlier, $k^x m \neq m$ for all $1 \leq x \leq p - 1$, so the orbit of $\chi_m$ under $\mathbb{Z}_p$ is $\{\chi_m, \chi_{km}, \chi_{k^2m}, \ldots, \chi_{k^{p-1}m}\}$. These orbits correspond to the sets $\mathcal{O}_m$ defined in section 3.3. As in the previous chapter, let us set $q := N / \text{gcd}(N, k - 1)$, and $\mathbb{Z}_N^\dagger := \{m \in \mathbb{Z}_N : m \neq qy \text{ for all } y \in \mathbb{Z}_N\}$. Choose some system of representatives $\chi_i$ from the orbits of $X$ under $\mathbb{Z}_p$. Recall that we have $ki = i$ exactly when $i \notin \mathbb{Z}_N^\dagger$. 
First consider the case where \( i \notin \mathbb{Z}_N^{\dag} \). Let \( \rho_j \in \text{Hom}(\mathbb{Z}_p, \mathbb{C}^*) \) for \( 0 \leq j \leq p - 1 \) be an irreducible representation of \( \mathbb{Z}_p \) defined by \( \rho_j(1) = \zeta^j \), where \( \zeta \) is a primitive \( p \)-th root of unity. Let \( \pi: G \to \mathbb{Z}_p \) be the canonical projection. Define

\[
\theta_{i,j} = \chi_i \otimes (\rho_j \circ \pi) \in \text{Hom}(G, \mathbb{C}^*).
\]

(See Definition A.8 for the definition of a tensor product of representations.) Then \( \theta_{i,j} \) is an irreducible one dimensional representation of \( G \). Explicitly, we have

\[
\theta_{i,j}(x, y) = (\chi_i \otimes (\rho_j \circ \pi))(x, y) = \chi_i(y) \cdot \rho_j(x) = \omega^{iy} \cdot \zeta^{ix}.
\]  

(4.1)

Now consider the case where \( i \in \mathbb{Z}_N^{\dag} \). Define

\[
\theta_{i,*} = \text{Ind}_{\mathbb{Z}_N}^G(\chi_i) \in \text{Hom}(G, GL_p(\mathbb{C})),
\]

(See Definition A.9 for the definition of a tensor product of representations.) Then \( \theta_{i,*} \) is an irreducible \( p \)-dimensional representation of \( G \). We can compute the character of \( \theta_{i,*} \) by applying the Frobenius formula. Let \( \hat{\chi}_i(g) = \chi_i(g) \) if \( g \in \mathbb{Z}_N \) and 0 otherwise. Then for \( (x, y) \in G \), we have

\[
\text{Tr}(\theta_{i,*}(x, y)) = \sum_{(x',0) \in G/\mathbb{Z}_N} \hat{\chi}_i((-x',0)(x,y)(x',0)) = \sum_{(x',0) \in G/\mathbb{Z}_N} \hat{\chi}_i(x,kx'y) = \left\{ \begin{array}{ll} 0 & : x \neq 0 \\ \sum_{j=0}^{p-1} \omega^{kjy} & : x = 0 \end{array} \right.
\]

Let \( g := (x, y) \) and \( \bar{s} := (s, 0) \) be elements of \( G \). We compute the matrix for \( \theta_{i,*} \) (see Fact A.10 for an explanation of this computation):

\[
\theta_{i,*}(g) = \begin{pmatrix}
\hat{\chi}_i(0 \cdot g \cdot 0) & \hat{\chi}_i(0 \cdot g \cdot 1) & \cdots & \hat{\chi}_i(0 \cdot g \cdot p-1) \\
\hat{\chi}_i((-1) \cdot g \cdot 0) & \hat{\chi}_i((-1) \cdot g \cdot 1) & \cdots & \hat{\chi}_i((-1) \cdot g \cdot p-1) \\
\vdots & \vdots & \ddots & \vdots \\
\hat{\chi}_i((-p+1) \cdot g \cdot 0) & \hat{\chi}_i((-p+1) \cdot g \cdot 1) & \cdots & \hat{\chi}_i((-p+1) \cdot g \cdot p-1) \\
\hat{\chi}_i(x, k^0y) & \hat{\chi}_i(x+1, k^1y) & \cdots & \hat{\chi}_i(x+(p-1), k^{p-1}y) \\
\hat{\chi}_i(x-1, k^0y) & \hat{\chi}_i(x, k^1y) & \cdots & \hat{\chi}_i(x+(p-2), k^{p-1}y) \\
\vdots & \vdots & \ddots & \vdots \\
\hat{\chi}_i(x-(p-1), k^0y) & \hat{\chi}_i(x-(p-2), k^1y) & \cdots & \hat{\chi}_i(x, k^{p-1}y) \\
0 & \cdots & 0 & \omega^{k^{-1}iy} & 0 & \cdots & 0 \\
0 & \cdots & 0 & 0 & \omega^{k^{-1}iy} & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & \omega^{k^{-1}iy} & 0 & \cdots & 0 \\
\end{pmatrix}
\]

(4.2)

Mackey’s criterion guarantees that for any of the \( \theta_{i,j} \) or \( \theta_{i,*} \) we have just defined, the representation \( \theta_{i,j} \) is indeed irreducible and if \( \theta_{i,j} \) and \( \theta_{i',j'} \) are isomorphic, then
$i = i'$ and $j = j'$. Therefore these $(Np)/q$ one-dimensional representations and $(N - (N/q))/p$ $p$-dimensional representations are all the irreducible representations of $G$, as can be verified by the following calculation (see Fact A.5):

$$
\frac{N}{q} \times p \times 1 + \left( \frac{N - N}{q} \right) \times p^2 = np = |G|.
$$

In conclusion, we have proved the following:

**Theorem 4.4.** A complete set of irreducible representations of $\mathbb{Z}_N \ltimes_k \mathbb{Z}_p$ is $\theta_{i,*}$ for $i \in \mathbb{Z}_N^1$ and $\theta_{i,j}$ for $i \in \mathbb{Z}_N \setminus \mathbb{Z}_N^1$, $j \in \mathbb{Z}_p$, where the $\theta$ are defined as in (4.1) and (4.2) above.

### 4.3 The Fourier transform over $\mathbb{Z}_N \ltimes_k \mathbb{Z}_p$

In the algorithm presented in chapter 3, rather than use the Fourier transform over $G := \mathbb{Z}_N \ltimes_k \mathbb{Z}_p$, we applied a series of Fourier transforms on each register. In this section, we show that our application of $\mathbb{Z}_p$ followed by $\mathbb{Z}_N$ approximates $\mathcal{F}_G$ in a certain sense.

Suppose $\hat{G}$ is a complete set of the irreducible representations of $G$; we can suppose further that the elements of $\hat{G}$ are labeled as $\theta_{i,j}$ or $\theta_{i,*}$ as defined in the previous section. Let $\hat{\mathbb{Z}}_N^1$ be the set of $i \in \mathbb{Z}_N^1$ such that $\theta_{i,*} \in \hat{G}$. Then, let us see how $\mathcal{F}_G$ acts on an element $|(x, y)\rangle \in \mathbb{C}G$:

$$
\mathcal{F}_G (|x, y\rangle) = \sum_{\theta \in \hat{G}} \sum_{1 \leq a, b \leq d_p} \sqrt{\frac{d_\theta}{|\mathbb{C}|}} [\theta(x, y)]_{a, b} |\theta, a, b\rangle
$$

$$
= \frac{1}{\sqrt{pN}} \left( \sum_{i \in \mathbb{Z}_N \setminus \mathbb{Z}_N^1} \sum_{j=0}^{p-1} \theta_{i,j}(x, y) |\theta_{i,j}\rangle + \sqrt{2} \sum_{i \in \hat{\mathbb{Z}}_N^1} \sum_{1 \leq a, b \leq d_p} [\theta_{i,*}(x, y)]_{a, b} |\theta_{i,*}, a, b\rangle \right)
$$

$$
= \frac{1}{\sqrt{pN}} \left( \sum_{i \in \mathbb{Z}_N \setminus \mathbb{Z}_N^1} \sum_{j=0}^{p-1} \omega^{iy} \cdot \zeta^{jx} |\theta_{i,j}\rangle + \sqrt{2} \sum_{i \in \hat{\mathbb{Z}}_N^1} \sum_{j=0}^{2} \omega^{k^{-x} - jy} |\theta_{i,*}, j, j - x\rangle \right)
$$

$$
= \frac{1}{\sqrt{pN}} \left( \sum_{i \in \mathbb{Z}_N \setminus \mathbb{Z}_N^1} \sum_{j=0}^{p-1} \omega^{iy} \cdot \zeta^{jx} |j\rangle |i\rangle + \omega^{k^{-x}} \sqrt{2} \sum_{i \in \hat{\mathbb{Z}}_N^1} \omega^{iy} |0\rangle |i\rangle \right),
$$

where we have relabeled the vectors $|\theta_{i,j}\rangle$ as $|j\rangle |i\rangle \in \mathbb{C} \mathbb{Z}_p \otimes \mathbb{C} \mathbb{Z}_N$ and $|\theta_{i,*}, j, j - x + j\rangle$ as $|0\rangle |k^j i\rangle \in \mathbb{C} \mathbb{Z}_p \otimes \mathbb{C} \mathbb{Z}_N$.

On the other hand, $\mathcal{F}_p \otimes I$ acts on $|(x, y)\rangle \in \mathbb{C}G$ by

$$
(\mathcal{F}_p \otimes I) (|x, y\rangle) = \frac{1}{\sqrt{p}} \sum_{s=0}^{p-1} \zeta^{sx} |s\rangle |y\rangle.
$$
Now suppose we measure the first register of \((F_p \otimes I) |(x, y)\rangle\) and in particular measure 0; the resulting state is merely \(|y\rangle\). If we now apply \(F_N\) and call the resulting vector \(|\phi\rangle\), we have

\[
|\phi\rangle = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} \omega^{iy} |i\rangle.
\]

Note that \(|\phi\rangle\) agrees with \(F_G |(x, y)\rangle\) in the coordinates corresponding to \(i \in \mathbb{Z}_N^\dagger\). If we project \(F_G |(x, y)\rangle\) into \(\mathbb{C}Z_p \otimes \mathbb{C}Z_N^\dagger\) and call the resulting vector \(|\psi\rangle\), we have

\[
|\psi\rangle = \frac{1}{\sqrt{|\mathbb{Z}_N^\dagger|}} \sum_{i \in \mathbb{Z}_N^\dagger} \omega^{iy} |i\rangle.
\]

Thus, as long as we project our state vector into some subspace of \(\mathbb{C}Z_N^\dagger\), the procedure of applying \(F_p\), measuring the first register, and then applying \(F_N\) yields the same result as applying \(F_G\) would. We follow exactly this series of steps in the single-query algorithm of section 3.4. The vector \(|\psi^\mu\rangle\) defined in that section is created by applying \(F_p\) to the coset state \(|gH\rangle\) from the Standard Method, measuring 0 in the first register, applying \(F_N\) to the resulting vector, and finally projecting the state into some subspace of \(\mathbb{C}Z_N^\dagger\). However, the most remarkable step of the algorithm, in which we sum the amplitudes of \(|\psi^\mu\rangle\) in the \(|\mu\rangle\) and \(|-k\mu\rangle\) components, is still not adequately explained by the considerations of this chapter. It would be extremely interesting if representation theory gave some \textit{a priori} justification for this step. We have been unable to find such a justification.
Appendix A

Basic Representation Theory

In this appendix we quickly recap the basic definitions and theorems from group representation theory used in chapter 4; for a in-depth treatment of the representation theory of finite groups, see Serre [20] or Dummit and Foote [6], chapter 15.

Definition A.1. A representation of a finite group $G$ is a group homomorphism $\rho: G \to GL(V)$, where $V$ is some vector space over a field $F$. We will always take $F$ to be $\mathbb{C}$ and $V$ to be finite-dimensional of dimension $d_\rho$; thus we will consider $\rho$ to be a map from $G$ to $GL_{d_\rho}(\mathbb{C})$ and from now on we do not state the assumption that our base field is $\mathbb{C}$. Fixing a basis of $V \cong \mathbb{C}^{d_\rho}$, we can realize $\rho(g)$ as a $d_\rho \times d_\rho$ matrix for each $g \in G$. We say that $d_\rho$ is the dimension of $\rho$.

Two representations, $\rho_1: G \to GL(V)$ and $\rho_2: G \to GL(W)$ are isomorphic if there is some isomorphism $\phi$ from $V$ to $W$ such that $\rho_1(g)(v) = \rho_2(g)(\phi(v))$ for all $g \in G, v \in V$.

Definition A.2. A subspace $W \subset V$ is said to be an invariant subspace of $\rho$ if $\rho(g)(W) \subset W$ for all $g \in G$. If there are no nontrivial (i.e. proper and nonzero) invariant subspaces of $\rho$, then we say $\rho$ is irreducible.

Definition A.3. Let $\rho: G \to GL(V)$ be a representation of $G$, and suppose we have $V \cong V_1 \oplus V_2$, where $V_1$ and $V_2$ are nontrivial invariant subspaces of $\rho$. Then $\rho$ is decomposable, and we write $\rho = \rho_1 \oplus \rho_2$, where $\rho_i$ is the restriction of $\rho$ to $V_i$. If $\rho$ is not decomposable, we say $\rho$ is indecomposable.

An important first result in representation theory says that irreducibility is the same as indecomposability as long as we are working over a field of characteristic 0:

Theorem A.4. (Maschke) A representation $\rho$ of $G$ is irreducible if and only if it is indecomposable. Thus any representation $\rho$ can be written as,

$$\rho = \rho_1 \oplus \cdots \oplus \rho_k,$$

where $\rho_1, \ldots, \rho_k$ are irreducible representations.
Because we can break up any reducible representation into a direct sum of irreducible representations, we focus our attention on irreducible representations, which we sometimes call irreps for shorthand. The following fact helps us find all the irreps:

**Fact A.5.** A finite group $G$ has a finite number of non-isomorphic irreducible representations. We call a maximal set of non-isomorphic irreps a complete set of irreps. Let $G$ be a complete set of irreps of $G$. Then,

$$|G| = \sum_{\rho \in G} d_{\rho}^2,$$

where $d_{\rho}$ is the dimension of $\rho$.

In section 4.1 we use the following corollary of Schur’s lemma (see [6], §15.1):

**Lemma A.6.** (Orthogonality of irreducible representations)

For two irreps $\rho, \rho'$ of $G$ we have

$$\frac{d_{\rho}}{|G|} \sum_{g \in G} [\rho(g)]_{i,j} [\rho'(g)]_{i',j'} = \delta_{\rho,\rho'} \delta_{i,i'} \delta_{j,j'},$$

where $\delta_{\rho,\rho'} = 1$ if $\rho = \rho'$ and 0 otherwise.

**Definition A.7.** Associated to each representation $\rho$ is its character, $\chi_{\rho}: G \to \mathbb{C}$, where, for each $g \in G$, $\chi_{\rho}(g) = \text{tr}(\rho(g))$, the matrix trace of $\rho(g)$. Note that this trace is independent of the basis of the matrix representation of $\rho(g)$. We can see that a one-dimensional representation is essentially the same as its character.

**Definition A.8.** If $\rho_1: G \to GL(V)$ and $\rho_2: G \to GL(W)$ are representations of $G$, then the tensor product of these representations is $\rho_1 \otimes \rho_2: G \to GL(V \otimes W)$, and it acts as we would expect: $(\rho_1 \otimes \rho_2)(g)(v \otimes w) = \rho_1(g)(v) \otimes \rho_2(g)(w)$. We can check that $\chi_{\rho_1 \otimes \rho_2} = \chi_{\rho_1} + \chi_{\rho_2}$ and $\chi_{\rho_1 \otimes \rho_2} = \chi_{\rho_1} \cdot \chi_{\rho_2}$.

**Definition A.9.** Let $\rho: H \to GL(V)$ be a representation of $H$, where $H$ is some subgroup of $G$. Let $t_1, \ldots, t_n$ be representatives of the cosets of $H$ in $G$. Then the induced representation $\text{Ind}_{H}^{G}(\rho)$ acts on the space $V = \bigoplus_{i=1}^{n} t_i V$ by $g \cdot \sum_{i}^{n} t_i v_i = \sum_{i}^{n} t_{j(i)} \pi(h_i) v_i$, where $v_i \in V$ and we have written $gx_i = x_j h$ with $h \in H$ in a unique way for each $i$.

The following fact lets us compute an explicit description of the induced representation:

**Fact A.10.** Let $\rho: H \to GL(V)$ be a representation of $H < G$. Define $\hat{\rho}: G \to GL(V)$ to be $\rho(g)$ if $g \in H$ and 0 otherwise. Let $t_1, \ldots, t_n$ be representatives from the cosets of $H$ in $G$. The matrix of the induced representation is the block matrix

$$\text{Ind}_{H}^{G}(\rho) = \begin{pmatrix} \hat{\rho}(t_1^{-1} g t_1) & \hat{\rho}(t_1^{-1} g t_2) & \cdots & \hat{\rho}(t_1^{-1} g t_n) \\ \hat{\rho}(t_2^{-1} g t_1) & \hat{\rho}(t_2^{-1} g t_2) & \cdots & \hat{\rho}(t_2^{-1} g t_n) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\rho}(t_n^{-1} g t_1) & \hat{\rho}(t_n^{-1} g t_2) & \cdots & \hat{\rho}(t_n^{-1} g t_n) \end{pmatrix}.$$
References


