Hidden Subgroup Problem

CompSci/Physics C191: Quantum Information Science and Technology

Sudhanva Kulkarni, Marnix Looijmans, Sameer Nayyar

May 2023

Abstract

The Hidden Subgroup Problem is a computational algebra problem with a wide range of applications. In this report, we discuss various algorithms to solve the HSP for different sorts of groups, namely Abelian, "almost Abelian", and non-Abelian finite groups. In addition to this, we discuss the applications of the HSP. The finite Abelian HSP can be applied to a large array of different problems, including computing the discrete logarithm, computing the order of a subgroup, and factoring integers. We show that most common quantum algorithms are special cases of the finite Abelian HSP, with particular choices of the group and subgroup. Finally, we discuss the applications of the non-Abelian HSP, including the Shortest Vector Problem and the Graph Automorphism Problem.

Statement of Contribution

The work on this report was distributed roughly equally between the group members. SK wrote the sections on QFT, intro to non-Abelian, solvable groups (similar problems section), nilpotent groups (almost abelian section), and dihedral groups. ML wrote the introduction, the section on applications of the Hidden Subgroup Problem, and the subsection on symmetric groups. SN wrote the abstract, the section on orthogonal subgroups, the section on the algorithm for the finite Abelian HSP, and the conclusion.

1 Introduction

In the 1980s, Richard Feynman came up with a proposal for reversible computation using the machinary of quantum mechanics [Fey82]. These so called quantum computers work with 'qubits', which consist of a superposition of 0s and 1s. Qubits can be manipulated by unitary transformations, which are fully invertible, and together qubits and unitary operators form the building blocks of quantum algorithms. [Vaz23]

Over the years various problems have been identified that can be more efficiently solved by a quantum algorithm than a classical algorithm. The first such example is the Deutsch algorithm from 1985 [Deu85]. More involved examples are the Bernstein-Vazirani problem and Simon's problem. [Vaz23]

One of the most significant developments in the field of quantum algorithms was the discovery of a polylogarithmic time algorithm for computing discrete logarithms and factoring large integers by Peter Shor in 1994 [Sho97]. Shor realized that both of these problems are in fact specific instances of the Hidden Subgroup Problem (HSP), which aided him in his work.

In this report we will review the Hidden Subgroup Problem. We will start with the problem statement and generalize the quantum Fourier transform from the lecture notes. Next, we will present the algorithm for solving the HSP for finite abelian groups.

Secondly, we will study specific instances of the HSP. We present Simon's algorithm and an algorithm for finding discrete logarithms and we will see their relation to the HSP for finite abelian groups. We will also include a table containing other specific instances of the HSP for reference. We then discuss groups that are "almost abelian".

In the final part of this report, we will review the Hidden Subgroup Problem for non-abelian groups. We will introduce the distinction between strong and weak Fourier sampling and following on from that, we will discuss dihedral groups and symmetric groups.

2 Problem Statement

Let G be a group and let H be a subgroup of G. Let $f : G \to X$ be a function such that for all $g_1, g_2 \in G$ we have that $f(g_1) = f(g_2)$ if and only if $g_1H = g_2H$, i.e., whenever g_1 and g_2 are in the same coset of H. We say that the function f "hides" the hidden subgroup H. Given an oracle that computes f on any arbitrary quantum state, our goal is to obtain a generating set of the subgroup H while use only $\text{poly}(\log |G|)$ queries to the oracle f.

This task is possible for finite Abelian groups, and for certain finite non-Abelian groups called solvable groups.

3 Quantum Fourier Transform over Groups

Here, we define the QFT over Abelian groups, which will be the most important piece of machinery we'll need for the algorithm. QFT over groups can be easily extrapolated from the QFT over $\mathbb{Z}_n$. There, we had $QFT\,|x\rangle \to \sum_{y\in\mathbb{Z}_n} \omega_n^{xy}\,|y\rangle$. To go about generalizing this, we must first take a slight detour into the representation of groups :

3.1 Representation of Groups

A **representation** of a **group** is essentially a mathematical object that captures how the group acts on a complex vector space V. More formally, it's some χ that takes elements of G to elements of

$GL(V)$. Further, we require that $\chi(g_1 + g_2) = \chi(g_1) \cdot \chi(g_2)$. Now, we can ask the following question - given some representation χ, is V the smallest non-trivial vector space on which χ works? In other words, is there a proper, **invariant subspace** $W \subset V$ such that χ, when restricted to W, is a valid representation? If the answer to the above question is no, we are dealing with an **irreducible representation**. The dimension of the smallest V is called the **dimension of the irreducible representation**. But why are these important? Let's consider a cyclic group to see why. Say we're dealing with $\mathbb{Z}_3$, and let's start simple - our vector space is simply $\mathbb{C}$ over itself. Now we can use a very familiar function for ψ - just take $\chi(1) = \omega_3$! It's easy to check that this satisfies our properties above. We can also pick $\chi(1) = \omega_3^2$. In fact, these are the only non-trivial representations for $\mathbb{Z}_3$ (The other is just $\chi(1) = 1$). One can show that in general, for $\mathbb{Z}_n$, the only irreducible representations are $\chi_k(1) = \omega_n^k$, where $k = 1, \ldots, n$.

3.2 Generalizing QFT

Now, using knowledge of the previous section, we can simply rewrite our expression for the QFT! Instead of $\text{QFT}\,|x\rangle \to \sum_{y \in \mathbb{Z}_n} \omega_n^{xy}\,|y\rangle$, we can just write $\text{QFT}\,|x\rangle \to \sum_{k \in \mathbb{Z}_n} \chi_k(x)\,|k\rangle$. Or more generally, if we have some G with representations $\hat{G}$, we get $\sum_{\chi \in \hat{G}} \chi(x)\,|\chi\rangle$ - And we can now use this for any G as long as we can find its representations!

3.3 Finite Abelian groups are good

We appeal to the following theorem without proof

Theorem 1 (Decomposition theorem) *Given any Abelian G, we can write it as a direct product of cyclic groups. So, $G = \mathbb{Z}_{p_1} \times \ldots \times \mathbb{Z}_{p_k}$.*

Each element of G can than be written as a $k-$tuple $(a_1, \ldots, a_k)$ (with each belong to their respective groups). The above theorem makes it so that we can simply multiply representations from each of the cyclic groups to get the representation of our abelian group! For instance, if we have $Z_3 \times Z_9$, we'd get $\psi_{k,l}(1,1) = \omega_3^k \omega_9^l\,|k,l\rangle$. In terms of quantum circuits, this corresponds to splitting our operator U as $U_1 \otimes U_2$, where the first one acts as $\mathbb{Z}_3$, and the other as $\mathbb{Z}_9$

4 Algorithm for solving the finite Abelian HSP

4.1 The Orthogonal Subgroup

Before describing the algorithm for solving the HSP for finite Abelian groups, we will discuss a few properties of characters of such groups and the QFT to make our later analysis simpler.

First, we will define $H^\perp$, the **orthogonal subgroup** to H as

$$H^\perp = \{g \in G : \chi_g(h) = 1, \forall h \in H\}$$

In other words, we can say that

$$H^\perp = \{g \in G : H \leq \ker \chi_g\}.$$

Now, we can define the **character of a subgroup** as

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

It is true that $\chi_g(H) = 1$ when $H \leq \ker \chi_g$ and 0 otherwise [Lom04]. Decompose the subgroup H into a product of cyclic groups $\mathbb{Z}/n_1\mathbb{Z} \times \cdots \times \mathbb{Z}/n_k\mathbb{Z}$ and consider

$$\chi_g(H) = \frac{1}{|H|}\sum_{h\in H}\chi_g(h) = \sum_{h_1\in\mathbb{Z}/n_1\mathbb{Z}} \cdots \sum_{h_k\in\mathbb{Z}/n_k\mathbb{Z}} \prod_{j=1}^{k} \omega_{n_j}^{h_j g_j} =$$

$$\left(\sum_{h_1\in\mathbb{Z}/n_1\mathbb{Z}} \omega_{n_1}^{h_1 g_1}\right) \times \cdots \times \left(\sum_{h_k\in\mathbb{Z}/n_k\mathbb{Z}} \omega_{n_k}^{h_k g_k}\right)$$

Now, if $\omega_{n_j}^{g_j} \neq 1$ for some j then one of the terms in this product will be the geometric series $\sum_{g_j\in\mathbb{Z}/n_j\mathbb{Z}} \left(\omega_{n_j}^{g_j}\right)^{h_j}$, which clearly equals 0. The only way for this not to happen is if $\omega_{n_j}^{g_j} = 1$ for all j. This would mean that $\chi_g(h) = 1$ for all $h \in H$, which would mean that $\chi_g(H) = \frac{|H|}{|H|} = 1$, which is the result we wanted to show.

The key implication of this result is that we can cancel non-trivial characters when we have a superposition over characters applied to a subgroup, i.e.

$$\sum_{g\in G} \chi_g(H) |g\rangle = \sum_{g\in H^\perp} |g\rangle .$$

4.2 The Algorithm

We will now describe in detail the algorithm for solving the HSP for any finite Abelian group G equipped with a function f that hides the subgroup H. [CvD10] [Lom04]

The algorithm proceeds as follows:

1. First, we place the state $|0\rangle$ into a register. This state represents the identity element, $e \in G$. We apply the QFT to this register. Since every character maps e to 1 (the identity in $\mathbb{C}^\times$), we will obtain a uniform superposition over every character $\chi_g \in \hat{G}$. We can think of this superposition as simply a uniform superposition over the entire group G, denoted as

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

2. Now, we will apply the oracle for the function f in a second register, yielding the state

$$\frac{1}{\sqrt{|G|}}\sum_{g\in G}|g\rangle |f(g)\rangle .$$

3. Next, we will measure the second register in the computational basis. This will collapse the state of the second register into some arbitrary element in the co-domain of f. Then, the state of the first register will also collapse into a superposition over all the elements in the corresponding coset of H. This is because the elements in a single coset of H uniquely forms the pre-image of any arbitrary single element in the co-domain of f. We can denote this uniformly randomly chosen coset of H as $s + H$, and we can describe the resulting state in the first register as

$$|s + H\rangle = \frac{1}{\sqrt{|H|}}\sum_{h\in H}|s + h\rangle .$$

4. Next, we will apply the QFT to the first register. The following calculations derive the final state:

$$\frac{1}{\sqrt{|H|}}\sum_{h\in H}\text{QFT }\ |s+h\rangle = \frac{1}{\sqrt{|G||H|}}\sum_{h\in H}\sum_{g\in G}\chi_g(s+h)\,|g\rangle = \sqrt{\frac{|H|}{|G|}}\sum_{g\in G}\chi_g(s+H)\,|g\rangle =$$

$$\sqrt{\frac{|H|}{|G|}}\sum_{g\in G}\chi_g(s)\chi_g(H)\,|g\rangle = \sqrt{\frac{|H|}{|G|}}\sum_{g\in H^\perp}\chi_g(s)\,|g\rangle\,.$$

   Here we have applied the definition of the QFT, as well as the previously proven property that $\chi_g(H)$ will equal 1 precisely when $g \in H^\perp$ and will equal 0 otherwise.

5. We will then measure the resulting state in the first register in the computational basis. The coefficients of all the states are precisely equal in magnitude, since $\chi_g(s)$ will always be a complex number on the unit circle, i.e., a number with magnitude 1. This means that there is an equal chance of the state collapsing to any one of the states in the superposition. So, when we measure this register, we will obtain a uniformly randomly chosen element of $H^\perp$.

6. It can be shown that if we repeat this process $O(\log|G|)$ times, we will, in expectation, obtain enough unique elements of $H^\perp$ to create a generating set for $H^\perp$. Furthermore, we know that since $H^\perp$ uniquely determines H, the generating set of $H^\perp$ uniquely determines the generating set of H. So, we can compute the generating set of the hidden subgroup H in a classical post-processing step. Since every iteration uses a single query to the oracle of f, we can see that the entire quantum algorithm has $O(\log|G|)$ query complexity.

5 Applications

In this section we will study examples of quantum algorithms that are special cases of the Hidden Subgroup Problem for Abelian groups. We previously discussed the general implementation of the Hidden Subgroup Problem and saw that it runs in polylogarithmic time. If we can therefore use the HSP framework we developed to solve specific problems such as factoring large integers or finding discrete logarithms, then the results for the HSP will also hold for these cases and we will be able to solve these problems in polylogarithmic time as well.

5.1 Simon's problem

We are given a function $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ such that $f(x) = f(y)$ if and only if $x \oplus y = s$, where $s \in \mathbb{Z}_2^n$ is an n-bit string. We are concerned with finding s. In this case, the parent group $G = \mathbb{Z}_2^n$, which is the group of all n-bit strings under the bitwise XOR operation. The algorithm proceeds as follows. [Vaz23]

1. We start with the state consisting of two registers $|0\rangle^{\otimes n}|0\rangle^{\otimes n}$ and apply the Hadamard transform $H^{\otimes n}$ on the first register to obtain a a superposition over all the elements in the group G. We then apply U_f in order to query f on the second register and we obtain

$$\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle\,|f(x)\rangle\,.$$

   where $N = 2^n$.

2. Next, we measure the second register which will cause the first register to collapse into
$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_0 \oplus s\rangle),$$
since x_0 and $x_0 \oplus s$ are the only pre-images of $f(x_0)$. We see that $\{x_0, x_0 \oplus s\} = x_0 \oplus H$ is a coset of the group $H = \{0, s\} \cong \mathbb{Z}_2$. The group H is a subgroup of G.

3. Finally, we apply the Hadamard transform again and we obtain
$$\frac{1}{\sqrt{2N}} \sum_{y=0}^{N-1} (-1)^{x_0 \cdot y} (1 + (-1)^{s \cdot y}) |y\rangle$$
When we make a measurement, we are guaranteed to get a value y such that $y \cdot s = 0$.

We repeat this process until we have $n - 1$ linearly independent values $y_1, y_2, ..., y_{n-1}$ so that we can solve for the n-bit string s. The hidden subgroup in this algorithm is $H = \{0, s\}$.

5.2 Discrete logarithms

Suppose we are given an element $x \in \mathbb{Z}_p^\times$, where p is prime. If g is a generator of $\mathbb{Z}_p^\times$, then we can uniquely write $x = g^l \bmod p$ for some $l \in \mathbb{Z}_N$, where we defined $N = p - 1$. The value $l = \log_g(x)$ is called the discrete logarithm of x with respect to g.

We define a function $f : \mathbb{Z}_N \times \mathbb{Z}_N \to \mathbb{Z}_p^\times$ given by $f(a, b) = g^a x^b \bmod p$. In this case $G = \mathbb{Z}_N \times \mathbb{Z}_N$ under the operation of addition modulo N. Observe that we can rewrite this function as $f(a, b) = g^{a+lb} \bmod p$. The algorithm to determine $l = \log_g(x)$ proceeds as follows. [CvD10]

1. We initialize the state $|0\rangle^{\otimes N} |0\rangle^{\otimes N} |0\rangle^{\otimes N}$ consisting of three registers. We apply Quantum Fourier Transform on the first two registers to obtain a superposition over all the elements of the group G. We then query the function f in the third register by applying U_f to the state. The result is
$$\frac{1}{N} \sum_{a=0}^{N-1} \sum_{b=0}^{N-1} |a\rangle |b\rangle |f(a, b)\rangle .$$

2. Next, we measure the third register, which will cause the state to collapse to a superposition of states $|a\rangle |b\rangle$ with $f(a, b)$ constant, in other words with $a + lb = \gamma$ constant. The resulting state is
$$\frac{1}{\sqrt{N}} \sum_{b=0}^{N-1} |\gamma - lb\rangle |b\rangle .$$

3. Finally, we apply Quantum Fourier Transform again to both registers to obtain
$$\frac{1}{N^{3/2}} \sum_{b=0}^{N-1} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \omega_N^{u(\gamma - lb) + vb} |u\rangle |v\rangle = \frac{1}{\sqrt{N}} \sum_{u=0}^{N-1} \omega_N^{u\gamma} |u\rangle |lu\rangle ,$$
where ω_N is the N-th root of unity and where we used the fact that
$$\frac{1}{N} \sum_{b=0}^{N-1} \omega_N^{b(v - lu)} = \begin{cases} 1 \text{ if } v = lu, \\ 0 \text{ if } v \neq lu. \end{cases}$$
This is a superposition over all elements of the group $H = \{(u, lu) : u \in \mathbb{Z}_N\} \cong \mathbb{Z}_N$, which is a subgroup of G. When we measure the two registers, we obtain a pair (u, lu).

We repeat this process until we obtain a second pair (u', lu') for which $\gcd(u, u') = 1$. By Bezout's identity, there exist $\lambda, \lambda' \in \mathbb{Z}$ such that $u\lambda + u'\lambda' = 1$. Using this we can compute $lu\lambda + lu'\lambda' = l = \log_g(x)$ as desired. The hidden subgroup in this algorithm is $H = \{(u, lu) : u \in \mathbb{Z}_N\}$.

5.3 Summary of applications

There are many more examples of specific cases of the Hidden Subgroup Problem. A number of common problems that can be solved by a quantum algorithm, including the ones that we covered above, are shown in the table below. [Wan10] [CvD10]

Problem	G	H	f	**Comment**
Simon	$\mathbb{Z}_2^n$	$\{0, s\} \cong \mathbb{Z}_2$	$f(x) = f(y) \Leftrightarrow x \oplus y = s$	Want to find n-bit s.
Period finding over $\mathbb{Z}_N$	$\mathbb{Z}_N$	$r\mathbb{Z}_N$	$f(x) = f(y) \Leftrightarrow (x - y)/r \in \mathbb{Z}$	Want to find period r.
Period finding over $\mathbb{Z}$	$\mathbb{Z}_N$	$r\mathbb{Z}_N$	$f(x) = f(y) \Leftrightarrow (x - y)/r \in \mathbb{Z}_N$	Want to find period r. Ensure $r << N$.
Deutsch	$\{0, 1\}$	$\{0, 1\}$ or $\{0\}$ depending on f	$f(0) = f(1)$ or $f(0) \neq f(1)$	Want to determine the behavior of f
Discrete logarithms	$\mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}$	$(\log_g x, 1)\mathbb{Z}_{p-1}$	$f(a, b) = g^a x^b \bmod p$	Want to find l such that $g^l = x \bmod p$

Other important problems that can be solved by a quantum algorithm are order finding over $\mathbb{Z}_N$ and factoring large integers. Order finding over $\mathbb{Z}_N$ is a special case of period finding over $\mathbb{Z}_N$ where $f(x) = a^x \bmod N$. In this case, we want to find the order r of $a \bmod N$. [Wan10]

Similarly, factoring large integers is a special case of order finding. This is referred to as Shor's algorithm. Suppose we wish to factor an integer N. We pick a random number $a \in \{2, 3, ..., N-1\}$. If $\gcd(N, a) \neq 1$, we are done, otherwise, we continue. We then determine the order r of $a \bmod N$. If r is odd, we start again. If r is even, we compute $\gcd(a^{r/2} - 1, N)$ and $\gcd(a^{r/2} + 1, N)$. If these are non-trivial factors of N, then we are done, otherwise we start again. We can therefore factor N in a couple of runs of the order finding algorithm. [Vaz23]

The ability to efficiently compute discrete logarithms or factor large integers can break many encryption protocols. For example, the Diffie-Hellman Key Exchange and RSA depend on the fact that computing discrete logarithms or factoring large integers, respectively, are classically hard problems. If we manage to implement the quantum algorithm to solve the Hidden Subgroup Problem, then we can break these protocols. [CvD10]

6 "Almost Abelian" Groups

Before diving deep into the non-Abelian side, we look at an "almost Abelian" groups. Two close relatives of Abelian groups are solvable groups and nilpotent groups. The motivation for the above comes from the commutator operation. For any $g, h \in G$(a group) we define $[g, h]$ as $g^{-1}h^{-1}gh$. Accordingly, we define $[G, G]$ as the set of all $[g, h]$ for every $g, h \in G$. Notice how $[G, G] = \{e\}$ for an Abelian group! A solvable group in this sense is essentially a more patient aAelian group. Rather than terminating $\{e\}$ instantly, a solvable group is one where the sequence $G_{n+1} = [G_n, G_n]$ with $G_0 = G$ terminates in $\{e\}$ after a finite number of iterations rather than strictly instantly. A nilpotent group similarly has a sequence of normal subgroups G_i such that $G_{i+1} = [G, G_i]$, and this terminates in $\{e\}$ finitely. If the sequence terminates at or before G_n, we call the group a nil-n group.

The algorithm (presented in [ISS07]) we'll look at in particular will be the hidden subgroup problem for all nil-2 groups G such that $|G| = p^\alpha$ for p prime $\alpha \in \mathbb{N}$ and $g^p = e$ for every $g \in G$. (We'll save solvable groups for the last section where we look at a very similar problem of finding the order of groups) The paper proves the following nice result-

Theorem 2 *If f hides H with $|H| = p$ and G as above, we can find H if we can efficiently hide $H[G, G]$*

The proof is quite elegant and indirectly relies on relies on **Lagrange's theorem** which states that $|H|$ divides $|G|$.

Proof 1 *First, notice that $H[G, G]$ is indeed Abelian since H is cyclic (Abelian property seen by Lagrange's theorem) and G is nil-2. So noticing that $H \subseteq H[G, G]$ lets us use the Abelian algorithm on H as a subgroup of $H[G, G]$. But how do we find $H[G, G]$?*
Say $G/[G, G] \approx \mathbb{Z}_p^m$ and $[G, G] \approx \mathbb{Z}_p^d$ (note that we can only have these isomorphisms since $[G, G]$ must be Abelian and its order must divide p). So, we can write elements of G as $\prod_i x_i^{t_i} \prod_i z_i^{y_i}$, where $x_i \in \bar{G} \approx G/[G, G]$ and $z_i \in [G, G]$. $\bar{G}$ is essentially just the quotient group. In more informal terms, if $g = \prod_i x_i^{t_i} \prod_i z_i^{y_i}$, then $\bar{g} = \prod_i x_i^{t_i}$ and multiplication in $\bar{G}$ is defined by $\bar{g_1} * \bar{g_2} = \bar{g_1 g_2}$. Now, notice that $H[G, G] \cap \bar{G}$ is a subgroup of $(\bar{G}, *)$! But by the isomorphism above, and the decomposition theorem for Abelian groups,$\bar{G}$ is Abelian, and we can easily find $H[G, G] \cap \bar{G}$ using the Abelian algorithm! Finally, using the fact that , $H[G, G] = (H[G, G] \cap \bar{G})\bar{G}$ we get the group we desired.
Something we didn't mention in the proof is isolating $H[G, G]$. The superposition of elements in $H[G, G]$ is obtained using our hiding procedure.

Further, it was shown in [ISS07] that given f that hides H, we can construct a procedure to hide $H[G, G]$, but that proof requires an extensive background in group theory so we omit it here.

7 Non-Abelian Hidden Subgroup Problem

Before we generalize further, we must talk modify our machinery a bit more. We just modified our QFT to account for one-dimensional representations above. But now we need to make it even more general. Recall that for some representation χ, its range is $GL(V)$. If V is not one dimensional, we'll map to some invertible matrix. So, to define QFT over general groups, we just account for this fact by using the states $|\chi, i, j\rangle$ (where i, j represent the indices) as basis vectors, and $\chi(g)_{i,j}$ as coefficients. The state is of course normalized accordingly . Here, we distinguish between **strong** and **weak** Fourier sampling. The former measures the indices as well, but the latter only measures

the label of the representation χ. In general, the problem is much harder for the non-Abelian case and presents difficulty in both the complexity of the algorithm and efficient measurements. In fact, the symmetric group resists even strong Fourier sampling as shown in [MRS05]. Further, in [AMR05], the authors were also able to construct instances of "almost Abelian" groups immune to Strong Fourier sampling (in the sense that even strong Fourier sampling is unable to distinguish between subgroups).
With strong sampling, we can describe a general algorithm as in [CS22]. There are primarily 3 steps

1. First, we proceed as in the Abelian case, we apply QFT on $|e, 0\rangle$ and apply our oracle to get $\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g, f(g)\rangle$ (note that we have this simple form since $\chi(e)$ is always identity)
2. Now, measure our function value to get $\sum_{h \in H} |ch\rangle$.
3. Finally, we apply QFT to get $\sum_{\chi \in \hat{G}} \sqrt{\frac{dim_\chi}{|G||H|}} \sum_{i,j=1}^{dim_\chi} \chi(ch)_{i,j} |\chi, i, j\rangle$

We then pretty much just try our best to reconstruct H. For the Abelian case, we just measure $|\chi\rangle$ and we're done, but as mentioned above, sometimes even measuring all three of χ, i, j doesn't help us enough.

7.1 Dihedral groups

A dihedral group D_{2n} essentially describes the symmetries of a regular $n-$gon. In general, its elements take the form $s^j r^i$, where s is a reflection (so $s^2 = e$) and r is a rotation (so $r^n = e$) The HSP for dihedral groups (or DHSP) is of particular interest due to its connection to the Shortest Vector Problem. Unfortunately we don't have some crisp equivalence result for dihedral groups. The first sub-exponential time algorithm was described in [Kup04]. Kuperberg provided a geometric picture of the algorithm and reduced the problem of finding subgroups to determining the slope of reflections.

We head in a different direction and focus more on the representations of the dihedral group while using the general framework (as in [CS22]). If n is odd, we have 4 1D representations, and if n is even there are 2 of them. Further, we have $floor(\frac{n-1}{2})$ 2D representations in both cases. Now, we use the lemma that Kuperberg showed. They showed that the subgroup problem reduces to finding the slope of a reflection. In group theoretic terms, this reduces to finding $H_a = <sr^a>$. Now, let's see what happens when we apply the general algorithm above. unfortunately, if we get get $dim\chi = 2$, we are unable to differentiate between the subgroups since $P(\chi, i, j) = \frac{1}{|G|}$ in all cases. More explicitly, we have the following

$$\sum_{h \in H} \chi(r^\alpha h) = \begin{pmatrix} \omega_N^{\alpha k} & \omega_N^{-(a-\alpha)k} \\ \omega_N^{(a-\alpha)k} & \omega_N^{\alpha k} \end{pmatrix} = A$$

and,

$$\sum_{h \in H} \chi(sr^\alpha h) = \begin{pmatrix} \omega_N^{(a-\alpha)k} & \omega_N^{\alpha k} \\ \omega_N^{\alpha k} & \omega_N^{-(a-\alpha)k} \end{pmatrix}$$

Now, we can encode our first row as

$$\frac{1}{\sqrt{N}} \omega_N^{\alpha k} \otimes \frac{1}{\sqrt{2}} (|0\rangle + \omega_N^{\alpha k} |1\rangle)$$

Let $|\psi_k\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \omega_N^{\alpha k}|1\rangle)$ Now, one can check that

$$|\psi_p\rangle|\psi_q\rangle = \frac{1}{\sqrt{2}}(|\psi_{p+q}\rangle|0\rangle + \omega_N^{aq}|\psi_{p-q}\rangle|1\rangle)$$

Then, it is shown in [Kup11] (again by Kuperberg) that one can use such states to get the state $|0\rangle + (-1)^a|1\rangle$. This lets us estimate the parity of a, which again, Kuperberg shows is equivalent to solving dihedral HSP. Notice that here, it wasn't enough to just measure the label χ, we also had to indirectly use the indices by encoding our matrix. So this is indeed an instance of Strong sampling. the time and query complexity of the above algorithm is $2^{O(\sqrt{logN})}$
As a conclusion to the dihedral group chapter, we note the existence of optimal measurements for this particular instance of the HSP. Bacon, Childs and Dam in 2006 [BCvD06] that there exists a POVM known as a "pretty good measurement" that allows us to determine a with a probability of $\geq 1/8$ under the condition that we use m special samples and that $\frac{m}{log_2 N} > 1 + 4/log_2 N$. The "special samples" here are those of the form $\frac{1}{\sqrt{2}}(|0\rangle|\alpha\rangle + |1\rangle|a+\alpha\rangle)$.

7.2 Symmetric groups

The Hidden Subgroup Problem for symmetric groups is related to the graph automorphism problem [CvD10]. In the graph automorphism problem, the goal is to determine whether there exists a non-trivial automorphism of a graph. Let Γ be a graph of n vertices. The symmetric group S_n under function composition describes all possible reorderings of the n vertices of Γ. A reordering $\pi \in S_n$ is an automorphism of Γ if $\pi(\Gamma) = \Gamma$. The automorphisms of Γ form a subgroup $\mathrm{Aut}(\Gamma) \leq S_n$. We can study this problem in the framework of the HSP by considering the function $f : S_n \to \{\text{all graphs of } n \text{ vertices}\}$ given by $f(\pi) = \pi(\Gamma)$. Then $f(\pi_1) = f(\pi_2)$ if and only if $\pi_1(\Gamma) = \pi_2(\Gamma)$, i.e. if and only if $\Gamma = \pi_1^{-1}(\pi_2(\Gamma))$. This is equivalent to demanding that $\pi_1^{-1} \circ \pi_2 \in \mathrm{Aut}(\Gamma)$. In other words, $f(\pi_1) = f(\pi_2)$ if and only if π_1 and π_2 are in the same coset of $\mathrm{Aut}(\Gamma)$. We say that the function f "hides" the subgroup $\mathrm{Aut}(\Gamma)$ by symmetries. [DISW13]

The Hidden Subgroup Problem for symmetric groups is called the Hidden Symmetry Subgroup Problem (HSSP). An instance of the HSSP is specified by the group of all symmetries G, a finite set M, and a group action $\circ : G \times M \to M$. In the graph automorphism problem, we have $G = S_n$ and M is the set of all graphs of n vertices. We can see that the HSSP reduces to the HSP when $M = G$ and the group action is the group operation, i.e. $g \circ h = gh$ for $g, h \in G$. [DISW13]

8 Similar group theoretic problems

Generally, the HSP is used to study the structure of a given group by constructing its various subgroups. However, this isn't the only way one needs to go about studying groups. In [Wat01], groups are studied through an order-finding algorithm that uses an approach similar to the Abelian hidden subgroup problem. The algorithm is particularly for solvable groups. It exploits the fact that for a solvable group G, we can take the sequence for solvable groups described in section 6 and calculate the relative orders of each of the quotient groups formed i.e $r_i = |G_{i+1}/G_i| = |G_{i+1}|/|G_i|$. Then the order of G will end up being the product of all such r_i. In general, to calculate $|G|/|H|$, first initialize a register to the state $|H\rangle = \frac{1}{\sqrt{|H|}}\sum_{h\in H}|h\rangle$, then initialize some other A as a uniform superposition of elements in $\mathbb{Z}_n$ where n is very large. Then, we can use our regular HSP circuit - First we apply QFT (with $G = \mathbb{Z}_n$), then left multiply R by g^a (apply our function oracle), and finally apply the inverse of QFT.

We're then left with the state

$$\frac{1}{N}\sum_{a\in\mathbb{Z}_n}\sum_{b\in\mathbb{Z}_n}\omega_N^{-ab}\left|b\right\rangle\left|aH\right\rangle$$

Observing this gives us b, which, with high probability approximates kN/r! The rest of the procedure is similar to Shor's. We just keep measuring till we get co-primes and we can calculate r.
The final algorithm is then just a simple application of this order finding algorithm. Further intricacies in constructing the states $|G_i\rangle$ are described in [Wat01]. Watrous describes a subroutine to construct copies of $|\langle g\rangle H\rangle$ from copies of $|H\rangle$. This subroutine is then used with the period finding above which results in the following algorithm

1. Prepare $k(m+1)$ copies of the state $|H_0\rangle$, where $H_0 = \{e\}$.
2. for $j = 1, 2, \ldots, m$
   - Using $k-1$ of the copies of $|H_{j-1}\rangle$, compute r_j
   - Convert the remaining copies of $|H_{j-1}\rangle$ to $|H_j\rangle$
3. Output product of r_j

9 Conclusion

In this report, we have discussed the Hidden Subgroup Problem, algorithms for solving it in a variety of groups, and its applications. We can solve the Hidden Subgroup Problem for finite Abelian groups by using the General Quantum Fourier Transform, which we can compute by decomposing a given group into a product of cyclic groups. With this, we can also solve the HSP for some "almost Abelian" groups, such as solvable and nilpotent groups. Furthermore, we discussed the non-Abelian HSP, defining the Fourier Transform for non-Abelian groups in terms of group representations, considering both strong and weak Fourier sampling. We have shown that there are quantum algorithms to solve the HSP for dihedral groups in sub-exponential time, but no such algorithms have been found for symmetric groups.

We have discussed a variety of applications for both the Abelian and the non-Abelian HSP. Many historical speedups from quantum algorithms can be viewed as a special case of the finite Abelian HSP. Simon's algorithm, the period-finding algorithm, the Deutsch-Jozsa algorithm, and the discrete logarithm algorithm are all applications of the finite Abelian HSP with different choices of the group G, subgroup H, and oracle function f. With these algorithms, we can also build up the order-finding algorithm and Shor's algorithm for integer factoring. These special cases (specifically, the discrete logarithm algorithm and Shor's algorithm) can be applied to break a variety of cryptographic protocols, namely the Diffie-Hellman Key Exchange and RSA. The algorithm for the "almost Abelian" solvable groups also gives us an algorithm for order-finding. Further applications exist for the non-Abelian case; namely, the dihedral case lets us solve the Shortest Vector Problem and the symmetric case lets us solve the Graph Automorphism problem. Because the HSP can be applied to such a large variety of groups to solve such a wide array of problems, we can view it as the single problem that unifies most quantum algorithmic speedups.

References

[AMR05] Gorjan Alagic, Cristopher Moore, and Alexander Russell. Strong fourier sampling fails over g^n, 2005.

[BCvD06] Dave Bacon, Andrew M. Childs, and Wim van Dam. *Chicago Journal of Theoretical Computer Science*, 12(1):1–25, 2006.

[CS22] Imin Chen and David Sun. The dihedral hidden subgroup problem, 2022.

[CvD10] Andrew M. Childs and Wim van Dam. Quantum algorithms for algebraic problems. *Reviews of Modern Physics*, 82(1):1–52, jan 2010.

[Deu85] David Deutsch. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400(1818):97–117, July 1985.

[DISW13] Thomas Decker, Gá bor Ivanyos, Miklos Santha, and Pawel Wocjan. Hidden symmetry subgroup problems. *SIAM Journal on Computing*, 42(5):1987–2007, jan 2013.

[Fey82] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467–488, June 1982.

[ISS07] Gábor Ivanyos, Luc Sanselme, and Miklos Santha. An efficient quantum algorithm for the hidden subgroup problem in nil-2 groups, 2007.

[Kup04] Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem, 2004.

[Kup11] Greg Kuperberg. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem, 2011.

[Lom04] Chris Lomont. The hidden subgroup problem - review and open problems, 2004.

[MRS05] Cristopher Moore, Alexander Russell, and Leonard J. Schulman. The symmetric group defies strong fourier sampling: Part i, 2005.

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

[Vaz23] Umesh Vazirani. Compsci c191: Quantum information science and technology, lecture notes, February 2023.

[Wan10] Frédéric Wang. The hidden subgroup problem, 2010.

[Wat01] John Watrous. Quantum algorithms for solvable groups, 2001.

.