Quantum Computing Lecture Notes, Extra Chapter

Hidden Subgroup Problem

Ronald de Wolf

1 Hidden Subgroup Problem

1.1 Group theory reminder

A group G consists of a set of elements (which is usually denoted by G as well) and an operation $\circ : G^2 \to G$ (often written as addition or multiplication), such that

1. the operation is associative: $g \circ (h \circ k) = (g \circ h) \circ k$ for all $g, h, k \in G$;
2. there is an *identity element* $e \in G$ satisfying $e \circ g = g \circ e = g$ for every $g \in G$;
3. and every $g \in G$ has an *inverse* $g^{-1} \in G$, such that $g \circ g^{-1} = g^{-1} \circ g = e$ (if the group operation is written as addition, then g^{-1} is written as $-g$).

We often abbreviate $g \circ h$ to gh. The group is *Abelian* (or *commutative*) if $gh = hg$ for all $g, h \in G$. Simple examples of finite additive Abelian groups are $G = \{0,1\}^n$ with bitwise addition mod 2 as the group operation, and $G = \mathbb{Z}_N$, the "cyclic group" of integers mod N. The set $G = \mathbb{Z}_N^*$ is the multiplicative group consisting of all integers in $\{1, \ldots, N-1\}$ that are coprime to N, with multiplication mod N as the group operation.[1] An important example of a non-Abelian group is the "symmetric group" S_n, which is the group of $n!$ permutations of n elements, using composition as the group operation.

A *subgroup* H of G, denoted $H \leq G$, is a subset of G that is itself a group, i.e., it contains e and is closed under taking products and inverses. A (left) *coset* of H is a set $gH = \{gh \mid h \in H\}$, i.e., a translation of H by the element g. All cosets of H have size $|H|$, and it is easy to show that two cosets gH and $g'H$ are either equal or disjoint, so the set of cosets partitions G into equal-sized parts.[2] Note that g and g' are in the same coset of H iff $g^{-1}g' \in H$.

If $T \subseteq G$, then we use $\langle T \rangle$ to denote the set of elements of G that we can write as products of elements from T and their inverses. This $H = \langle T \rangle$ is a subgroup of G, and T is called a *generating set* of H. Note that adding one more element $t \notin \langle T \rangle$ to T at least doubles the size of the generated subgroup, because H and tH are disjoint and $H \cup tH \subseteq \langle T \cup \{t\} \rangle$. This implies that every $H \leq G$ has a generating set of size $\leq \log |H| \leq \log |G|$. We abbreviate $\langle \{\gamma\} \rangle$ to $\langle \gamma \rangle$, which is the cyclic group generated by γ; every cyclic group of size N is isomorphic to $\mathbb{Z}_N$.

[1] This group $\mathbb{Z}_N^*$ is isomorphic to the additive group $\mathbb{Z}_{\phi(N)}$, where Euler's ϕ-function counts the elements of $\{1, \ldots, N-1\}$ that are coprime to N. For example, $\mathbb{Z}_p^*$ is isomorphic to $\mathbb{Z}_{p-1}$.

[2] This also proves Lagrange's theorem for finite groups: if $H \leq G$ then $|H|$ divides $|G|$.

1.2 Definition and some instances of the HSP

The *Hidden Subgroup Problem* is the following:

> Given a known group G and a function $f : G \to S$ where S is some finite set.
> Suppose f has the property that there exists a subgroup $H \leq G$ such that f is constant within each coset, and distinct on different cosets: $f(g) = f(g')$ iff $gH = g'H$.
> Goal: find H.

We assume f can be computed efficiently, meaning in time polynomial in $\log |G|$ (the latter is the number of bits needed to describe an input $g \in G$ for f). Since H may be large, "finding H" typically means finding a generating set for H.

This looks like a rather abstract algebraic problem, but many important problems can be written as an instance of the HSP. We will start with some examples where G is Abelian.

Period-finding. As we saw in Chapter 5, we can factor a large number N if we can solve the following: given an x that is coprime to N and associated function $f : \mathbb{Z} \to \mathbb{Z}_N^*$ by $f(a) = x^a \bmod N$, find the period r of f.[3] Since $\langle x \rangle$ is a size-r subgroup of the group $\mathbb{Z}_N^*$, the period r divides $|\mathbb{Z}_N^*| = \phi(N)$. Hence we can restrict the domain of f to $\mathbb{Z}_{\phi(N)}$.

Period-finding is an instance of the HSP as follows. Let $G = \mathbb{Z}_{\phi(N)}$ and consider its subgroup $H = \langle r \rangle = \{0, r, 2r, \ldots, \phi(N) - r\}$ of all multiples of r up to $\phi(N)$. Note that because of its periodicity, f is constant on each coset $s + H$ of H, and distinct on different cosets. Also, f is efficiently computable by repeated squaring. Since the hidden subgroup H is generated by r, finding the generator of H solves the period-finding problem.

Discrete logarithm. Another problem often used in classical public-key cryptography is the discrete logarithm problem: given a generator γ of a cyclic multiplicative group C of size N (so $C = \{\gamma^a \mid a \in \{0, \ldots, N-1\}\}$), and $A \in C$, can we find the unique $a \in \{0, 1, \ldots, N-1\}$ such that $\gamma^a = A$? This a is called the discrete logarithm of A (w.r.t. generator γ). It is generally believed that classical computers need time roughly exponential in $\log N$ to compute a from A (and one can actually *prove* this in a model where we can only implement group operations via some "black-box"). This assumption underlies for instance the security of Diffie-Hellman key exchange (where $C = \mathbb{Z}_p^*$ for some large prime p, see Exercise 2), as well as elliptic-curve cryptography.

Discrete log is an instance of the HSP as follows. We take $G = \mathbb{Z}_N \times \mathbb{Z}_N$ and define function $f : G \to C$ by $f(x, y) = \gamma^x A^{-y}$, which is efficiently computable by repeated squaring. For group elements $g_1 = (x_1, y_1), g_2 = (x_2, y_2) \in G$ we have

$$f(g_1) = f(g_2) \iff \gamma^{x_1 - a y_1} = \gamma^{x_2 - a y_2} \iff (x_1 - x_2) = a(y_1 - y_2) \iff g_1 - g_2 \in \langle (a, 1) \rangle.$$

Let H be the subgroup of G generated by the element $(a, 1)$, then we have an instance of the HSP. Finding the generator of the hidden subgroup H gives us a, solving the discrete log problem.

2 An efficient quantum algorithm if G is Abelian

In this section we show that HSPs where G (and hence H) is Abelian, and where f is efficiently computable, can be solved efficiently by a quantum algorithm. This generalizes Shor's factoring algorithm, and will also show that discrete logarithms can be computed efficiently.

[3] This r is also known as the *order* of the element x in the group $\mathbb{Z}_N^*$, so this problem is also known as *order-finding*.

2.1 Representation theory and the quantum Fourier transform

We start by explaining the basics of representation theory. The idea here is to replace group elements by matrices, so that linear algebra can be used as a tool in group theory. A d-dimensional *representation* of a multiplicative group G is a map $\rho : g \mapsto \rho(g)$ from G to the set of $d \times d$ invertible complex matrices, satisfying $\rho(gh) = \rho(g)\rho(h)$ for all $g, h \in G$. The latter property makes the map ρ a *homomorphism*. It need not be an *iso*morphism; for example, the constant-1 function is a trivial representation of any group. The *character* corresponding to ρ is the map $\chi_\rho : G \to \mathbb{C}$ defined by $\chi_\rho(g) = \mathrm{Tr}(\rho(g))$.

Below we restrict attention to the case where G is Abelian. In this case we may assume the dimension d to be 1 without loss of generality, so a representation ρ and the corresponding character χ_ρ are just the same function. Also, it is easy to see that the complex values $\chi_\rho(g)$ have modulus 1, because $|\chi_\rho(g^k)| = |\chi_\rho(g)|^k$ for all integers k. The "Basis Theorem" of group theory says that every finite Abelian group G is isomorphic to a direct product $\mathbb{Z}_{N_1} \times \cdots \times \mathbb{Z}_{N_k}$ of cyclic groups. First consider just one cyclic group $\mathbb{Z}_N$, written additively. Consider the discrete Fourier transform (Chapter 4), which is an $N \times N$ matrix. Ignoring the normalizing factor of $1/\sqrt{N}$, its kth column may be viewed as a map $\chi_k : \mathbb{Z}_N \to \mathbb{C}$ defined by $\chi_k(j) = \omega_N^{jk}$, where $\omega_N = e^{2\pi i/N}$. Note that $\chi_k(j + j') = \chi_k(j)\chi_k(j')$, so χ_k is actually a 1-dimensional representation (i.e., a character function) of $\mathbb{Z}_N$. In fact, the N characters corresponding to the N columns of the Fourier matrix are *all* the characters of $\mathbb{Z}_N$. For Abelian groups G that are (isomorphic to) a product $\mathbb{Z}_{N_1} \times \cdots \times \mathbb{Z}_{N_k}$ of cyclic groups, the $|G| = N_1 \cdots N_k$ characters are just the products of the characters of the individual cyclic groups $\mathbb{Z}_{N_j}$. Note that the characters are pairwise orthogonal.

The set of all characters of G forms a group $\widehat{G}$ with the operation of pointwise multiplication. This is called the *dual group* of G. If $H \leq G$, then the following is a subgroup of $\widehat{G}$ of size $|G|/|H|$:

$$H^\perp = \{\chi_k \mid \chi_k(h) = 1 \text{ for all } h \in H\}.$$

Let us interpret the quantum Fourier transform in terms of the characters. For $k \in \mathbb{Z}_N$, define the state whose entries are the (normalized) values of χ_k:

$$|\chi_k\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \chi_k(j)|j\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{jk}|j\rangle.$$

With this notation, the QFT just maps the standard basis of $\mathbb{C}^N$ to the orthonormal basis corresponding to the characters:

$$F_N : |k\rangle \mapsto |\chi_k\rangle.$$

As we saw in Chapter 4, this map can be implemented by an efficient quantum circuit if N is a power of 2. The QFT corresponding to a group G that is isomorphic to $\mathbb{Z}_{N_1} \times \cdots \times \mathbb{Z}_{N_k}$ is just the tensor product of the QFTs for the individual cyclic groups. For example, the QFT corresponding to $\mathbb{Z}_2$ is the Hadamard gate H, so the QFT corresponding to $\mathbb{Z}_2^n$ is $H^{\otimes n}$ (which is of course very different from the QFT corresponding to $\mathbb{Z}_{2^n}$).

2.2 A general algorithm for Abelian HSP

The following is an efficient quantum algorithm for solving the HSP for some Abelian group G (written additively) and function $f : G \to S$. This algorithm, sometimes called the "standard

algorithm" for HSP, was first observed by Kitaev [13] (inspired by Shor's algorithm) and worked out further by many, for instance Mosca and Ekert [16].

1. Start with $|0\rangle|0\rangle$, where the two registers have dimension $|G|$ and $|S|$, respectively.
2. Create a uniform superposition over G in the first register: $\frac{1}{\sqrt{|G|}}\sum_{g\in G}|g\rangle|0\rangle$.
3. Compute f in superposition: $\frac{1}{\sqrt{|G|}}\sum_{g\in G}|g\rangle|f(g)\rangle$.
4. Measure the second register. This yields some value $f(s)$ for unknown $s \in G$. The first register collapses to a superposition over the g with the same f-value as s (i.e., the coset $s+H$): $\frac{1}{\sqrt{|H|}}\sum_{h\in H}|s+h\rangle$.
5. Apply the QFT corresponding to G to this state, giving $\frac{1}{\sqrt{|H|}}\sum_{h\in H}|\chi_{s+h}\rangle$.
6. Measure and output the resulting g.

The key to understanding this algorithm is to observe that step 5 maps the uniform superposition over the coset $s+H$ to a uniform superposition over the labels of $H^\perp$:

$$\begin{aligned}\frac{1}{\sqrt{|H|}}\sum_{h\in H}|\chi_{s+h}\rangle &= \frac{1}{\sqrt{|H||G|}}\sum_{h\in H}\sum_{g\in G}\chi_{s+h}(g)|g\rangle \\ &= \frac{1}{\sqrt{|H||G|}}\sum_{g\in G}\chi_s(g)\sum_{h\in H}\chi_h(g)|g\rangle = \sqrt{\frac{|H|}{|G|}}\sum_{g:\chi_g\in H^\perp}\chi_s(g)|g\rangle,\end{aligned}$$

where the last equality follows from the orthogonality of characters of the group H (note that χ_g restricted to H is a character of H, and it's the constant-1 character iff $\chi_g \in H^\perp$):

$$\sum_{h\in H}\chi_h(g) = \sum_{h\in H}\chi_g(h) = \begin{cases} |H| & \text{if } \chi_g \in H^\perp \\ 0 & \text{if } \chi_g \notin H^\perp \end{cases}$$

The phases $\chi_s(g)$ don't affect the probabilities of the final measurement, since $|\chi_s(g)|^2 = 1$. The above algorithm thus samples uniformly from the (labels of) elements of $H^\perp$. Each such element $\chi_g \in H^\perp$ gives us a constraint on H because $\chi_g(h) = 1$ for all $h \in H$.[4] Generating a small number of such elements will give sufficient information to find the generators of H itself. Consider our earlier examples of Abelian HSP:

Period-finding. For the HSP corresponding to period-finding, $G = \mathbb{Z}_{\phi(N)}$ and $H = \langle r\rangle$, and

$$H^\perp = \{\chi_b \mid e^{2\pi i bh/\phi(N)} = 1 \text{ for all } h \in H\} = \{\chi_b \mid br/\phi(N) \in \{0,\ldots,r-1\}\}.$$

[4]This is a *linear* constraint mod N. Suppose for example that $G = \mathbb{Z}_{N_1}\times\mathbb{Z}_{N_2}$, $g = (g_1, g_2)$ is the label of an element of $H^\perp$. Then $1 = \chi_g(h) = \omega_{N_1}^{g_1h_1}\omega_{N_2}^{g_2h_2}$ for all $h = (h_1, h_2) \in H$, equivalently $g_1h_1N_2 + g_2h_2N_1 = 0 \bmod N$.

Accordingly, the output of the algorithm is an integer multiple $b = c\phi(N)/r$ of $\phi(N)/r$, for uniformly random $c \in \{0, \ldots, r-1\}$.

Notice that the algorithm doesn't actually know $\phi(N)$, which creates two problems. First, of the 4 numbers $b, c, \phi(N), r$ involved in the equation $b = c\phi(N)/r$ we only know the measurement outcome b, which is not enough to compute r. Second, step 5 of the algorithm wants to do a QFT corresponding to the group $\mathbb{Z}_{\phi(N)}$ but it doesn't know $\phi(N)$ (and even if we knew $\phi(N)$, we've only seen how to efficiently do a QFT over $\mathbb{Z}_q$ when q is a power of 2). Fortunately, if we actually use the QFT over $\mathbb{Z}_q$ for q a power of 2 that is roughly N^2 (and in step 1 set up a uniform superposition over $\mathbb{Z}_q$ instead of G), then one can show that the above algorithm still works, with high probability yielding a number b that's close to an integer multiple of q/r.[5] This is basically just Shor's algorithm as described in Chapter 5.

Discrete logarithm. For the HSP corresponding to the discrete log problem, where $G = \mathbb{Z}_N \times \mathbb{Z}_N$ and $H = \langle (a, 1) \rangle$, a small calculation shows that $H^\perp = \{\chi_{(c,-ac)}\}$. Hence sampling from $H^\perp$ yields some label $(c, -ac) \in G$ of an element of $H^\perp$, from which we can compute the discrete logarithm a. The QFT corresponding to G is $F_N \otimes F_N$, which we don't know how to implement efficiently, but which we can replace by $F_q \otimes F_q$ for some power-of-2 q somewhat larger than N.

In the above algorithm we assumed G is a *finite* Abelian group. These techniques have been much extended to the case of infinite groups such as $G = \mathbb{Z}$ and even $\mathbb{R}^d$, to obtain efficient quantum algorithms for problems like Pell's equation [9], and computing properties in number fields [4].

3 General non-Abelian HSP

3.1 The symmetric group and the graph isomorphism problem

The Abelian HSP covers a number of interesting computational problems, including period-finding and discrete log. However, there are also some interesting computational problems that can be cast as an instance of HSP with a *non-Abelian* G. Unfortunately we do not have an efficient algorithm for most non-Abelian HSPs.

A good example is the *graph isomorphism* (GI) problem: given two undirected n-vertex graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, decide whether there exists a bijection taking the vertices of $\mathcal{G}_1$ to those of $\mathcal{G}_2$ that makes the two graphs equal. No efficient classical algorithm is known for GI, so it would be great if we could solve this efficiently on a quantum computer.[6]

How can we try to solve this via the HSP? Let $\mathcal{G}$ be the $2n$-vertex graph that is the disjoint union of the two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$. Let $G = S_{2n}$. Let f map $\pi \in S_{2n}$ to $\pi(\mathcal{G})$, which means that edge (i, j) becomes edge $(\pi(i), \pi(j))$. Let H be the automorphism group $\mathrm{Aut}(\mathcal{G})$ of $\mathcal{G}$, which is the set of all $\pi \in S_{2n}$ that map $\mathcal{G}$ to itself. This gives an instance of the HSP, and solving it would give us a generating set of $H = \mathrm{Aut}(\mathcal{G})$.

Assume for simplicity that each of $\mathcal{G}_1$ and $\mathcal{G}_2$ is connected. If $\mathcal{G}_1$ and $\mathcal{G}_2$ are not isomorphic, then the only automorphisms of $\mathcal{G}$ are the ones that permute vertices inside $\mathcal{G}_1$ and inside $\mathcal{G}_2$:

[5] There is something to be proved here, but we will skip the details. In fact one can even use a Fourier transform for $q = O(N)$ instead of $O(N^2)$ [8]. Note that this also reduces the number of qubits used by Shor's algorithm from roughly $3 \log N$ to roughly $2 \log N$.

[6] For a long time, the best algorithm for GI took time roughly $2^{\sqrt{n}}$ [2], but in a recent breakthrough Babai gave a "quasi-polynomial" algorithm, which is $n^{(\log n)^{O(1)}}$ [1]. That's not yet polynomial time, but is a lot faster than before.

$\mathrm{Aut}(\mathcal{G}) = \mathrm{Aut}(\mathcal{G}_1) \times \mathrm{Aut}(\mathcal{G}_2)$. However, if the two graphs are isomorphic, then $\mathrm{Aut}(\mathcal{G})$ will also contain a permutation that swaps the first n with the second n vertices. Accordingly, if we were able to find a generating set of the hidden subgroup $H = \mathrm{Aut}(\mathcal{G})$, then we can just check whether all generators are in $\mathrm{Aut}(\mathcal{G}_1) \times \mathrm{Aut}(\mathcal{G}_2)$ and decide graph isomorphism.

3.2 Non-Abelian QFT on coset states

One can try to design a quantum algorithm for general, non-Abelian instances of the HSP along the lines of the earlier standard algorithm: set up a uniform superposition over a random coset of H, apply the QFT corresponding to G, measure the final state, and hope that the result gives useful information about H. QFTs corresponding to non-Abelian G are much more complicated than in the Abelian case, because the representations ρ can have dimension $d > 1$ and hence do not coincide with the corresponding character χ_ρ. For completeness, let's write down the QFT anyway. Let $\widehat{G}$ denote the set of "irreducible" representations of G, and $\dim(\rho)$ be the dimension of a particular $\rho \in \widehat{G}$. We can assume without loss of generality that the $\dim(\rho) \times \dim(\rho)$ matrices $\rho(g)$ are unitary. The QFT corresponding to G is defined as follows:

$$|g\rangle \longmapsto \sum_{\rho \in \widehat{G}} \sqrt{\frac{\dim(\rho)}{|G|}} |\rho\rangle \sum_{i,j=1}^{\dim(\rho)} \rho(g)_{ij} |i,j\rangle,$$

where $|\rho\rangle$ denotes a name or label of ρ. It can be shown that this map is unitary. In particular, $|G| = \sum_{\rho \in \widehat{G}} \dim(\rho)^2$, which implies that the dimensions on the left and the right are the same, and that the right-hand state has norm 1. In many cases this QFT can still be implemented with an efficient quantum circuit, including for the symmetric group $G = S_{2n}$ that is relevant for graph isomorphism [3, 14]. However, that is not enough for an efficient algorithm: the standard algorithm does not always yield much information about the hidden $H \leq S_{2n}$ [7, 15, 10].

There are some special cases of non-Abelian HSP that can be computed efficiently, for instance for normal subgroups [11], solvable groups [17], and nil-2 groups [12].

3.3 Query-efficient algorithm

While we do not have a general efficient quantum algorithm for the non-Abelian HSP, there does exist an algorithm that needs to compute f only a few times, i.e., a *query-efficient* algorithm. We will sketch this now. Consider steps 1–4 of the standard algorithm for the Abelian case. Even in the general non-Abelian case, this produces a coset state, i.e., a uniform superposition over the elements of a uniformly random coset of H. Suppose we do this m times, producing a state $|\psi_H\rangle$ which is the tensor product of m random coset states.[7] One can show that the states corresponding to different hidden subgroups are pairwise almost orthogonal: $|\langle \psi_H | \psi_{H'} \rangle|$ is exponentially small in m. The hidden subgroup H is generated by a set of $\leq \log |G|$ elements. Hence the total number of possible H that we want to distinguish is at most $\binom{|G|}{\log |G|} \leq 2^{(\log |G|)^2}$. This allows us to define a POVM $\{E_H\}$, with one element for each possible hidden subgroup H, such that if we measure $|\psi_H\rangle$ with this POVM, then we are likely to get the correct outcome H (see Exercise 3 for the idea). Choosing m some polynomial in $\log |G|$ suffices for this. While this POVM need not be efficiently

[7] Strictly speaking we should consider the tensor product of m copies of the *mixed* state ρ_H that is the uniform average over all coset states of H.

implementable, at least the number of times we need to query the function f is only m. Ettinger et al. [6] even showed that $m = O(\log |G|)$ suffices.

For those interested in more HSP results, a good source is Childs's lecture notes [5, Chapter 4–14].

Exercises

1. Show that the Deutsch-Jozsa problem for $n = 1$ (i.e., where $f : \{0,1\} \to \{0,1\}$) is an instance of the HSP. Explicitly say what G, f, H, and $H^\perp$ are, and how sampling from $H^\perp$ allows you to solve the problem.

2. This exercise explains Diffie-Hellman key exchange, which is secure under the assumption that the adversary cannot efficiently compute discrete logarithms. Alice and Bob choose a public key consisting of a large prime p (say, of 1000 or 2000 bits) and generator γ of the group $\mathbb{Z}_p^*$, which has size $\phi(p) = p - 1$. To agree on a shared secret key K, Alice chooses a uniformly random $a \in \{0, \ldots, p-2\}$ and sends Bob the group element $A = \gamma^a$; Bob chooses a uniformly random $b \in \{0, \ldots, p-2\}$ and sends Alice $B = \gamma^b$. Alice and Bob use $K = \gamma^{ab}$ as their secret key, which they can use for instance to encrypt messages using a one-time pad.

   (a) Show that both Alice and Bob can efficiently compute K given the communication.

   (b) Show that computing K from the information that an adversary may have obtained from the public key and the channel (i.e., A, B, p and γ, but not a and b) requires computing a discrete logarithm.

3. Suppose we are given an unknown state $|\psi_i\rangle$ from a known set of K states $\{|\psi_j\rangle \mid j \in [K]\}$.

   (a) Suppose the states are pairwise orthogonal: $\langle\psi_j|\psi_k\rangle = \delta_{jk}$. Give a projective measurement that determines i with probability 1.

   (b) Suppose the states are pairwise *almost* orthogonal: $|\langle\psi_j|\psi_k\rangle| \ll 1/K^2$ for all distinct $j, k \in [K]$. Define $E_i = \frac{2}{3}|\psi_i\rangle\langle\psi_i|$. Show that $I - \sum_{i=1}^K E_i$ is positive semidefinite.

   (c) Under the same assumption as (b), give a POVM that determines i with success probability at least $2/3$.

4. Suppose we have an efficient algorithm to produce, from a given undirected n-vertex graph $\mathcal{G}$, the following n^2-qubit state, where the basis states correspond to $n \times n$ adjacency matrices:

$$a_{\mathcal{G}} \sum_{\pi \in S_n} |\pi(\mathcal{G})\rangle.$$

   Here $a_{\mathcal{G}}$ is a scalar that makes the norm equal to 1. Use this procedure to efficiently decide whether two given graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are isomorphic or not.

References

[1] L. Babai. Graph isomorphism in quasipolynomial time. In *Proceedings of 48th ACM STOC*, pages 684–697, 2016. arXiv:1512.03547.

[2] L. Babai and E. M. Luks. Canonical labeling of graphs. In *Proceedings of 15th ACM STOC*, pages 171–183, 1983.

[3] R. Beals. Quantum computation of Fourier transforms over symmetric groups. In *Proceedings of 29th ACM STOC*, pages 48–53, 1997.

[4] J-F. Biasse and F. Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In *Proceedings of 27th ACM-SIAM SODA*, pages 893–902, 2016.

[5] A. Childs. Lecture notes on quantum algorithms. Technical report, University of Maryland, 2017. Available at https://cs.umd.edu/ amchilds/qa/.

[6] M. Ettinger, P. Høyer, and M. Knill. The quantum query complexity of the hidden subgroup problem is polynomial. *Information Processing Letters*, 91(1):43–48, 2004. quant-ph/0401083.

[7] M. Grigni, L. Schulman, M. Vazirani, and U. Vazirani. Quantum mechanical algorithms for the nonabelian hidden subgroup problem. *Combinatorica*, 24(1):137–154, 2004. Earlier version in STOC'01.

[8] L. Hales and S. Hallgren. An improved quantum Fourier transform algorithm and applications. In *Proceedings of 41st IEEE FOCS*, pages 515–525, 2000.

[9] S. Hallgren. Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem. *Journal of the ACM*, 54(1):653–658, 2007. Earlier version in STOC'02.

[10] S. Hallgren, C. Moore, M. Roetteler, A. Russell, and P. Sen. Limitations of quantum coset states for graph isomorphism. *Journal of the ACM*, 57(6):34, 2010. Earlier version in STOC'06.

[11] S. Hallgren, A. Russell, and A. Ta-Shma. The hidden subgroup problem and quantum computation using group representations. *SIAM Journal on Computing*, 32(4):916–934, 2003. Earlier version in STOC'00.

[12] G. Ivanyos, L. Sanselme, and M. Santha. An efficient quantum algorithm for the hidden subgroup problem in nil-2 groups. *Algorithmica*, 62(1–2):480–498, 2012. arXiv:0707.1260.

[13] A. Yu. Kitaev. Quantum measurements and the Abelian stabilizer problem. quant-ph/9511026, 12 Nov 1995.

[14] C. Moore, D. N. Rockmore, and A. Russell. Generic quantum Fourier transforms. *ACM Transactions on Algorithms*, 2(4):707–723, 2006. quant-ph/0304064.

[15] C. Moore, A. Russell, and L. Schulman. The symmetric group defies strong Fourier sampling. *SIAM Journal on Computing*, 37(6):1842–1864, 2008. quant-ph/0501056+66. Earlier version in FOCS'05.

[16] M. Mosca and A. Ekert. The hidden subgroup problem and eigenvalue estimation on a quantum computer. In *Proceedings of 1st NASA QCQC conference*, volume 1509 of *Lecture Notes in Computer Science*, pages 174–188. Springer, 1998. quant-ph/9903071.

[17] J. Watrous. Quantum algorithms for solvable groups. In *Proceedings of 33rd ACM STOC*, pages 60–67, 2001.