Order Finding for Solvable Groups

October 19, 2022

Abstract

The problem of computing the order of a group, whether or not it is known to be abelian, was shown by Babai and Szemerédi in 1984 to be impossible to solve classically in polynomial-time. In his paper *Quantum Algorithms for Solvable Groups*, John Watrous illustrates a quantum polynomial-time algorithm for computing the order of solvable groups, a superset of abelian groups. Most group-theoretic quantum algorithms focus on solving the hidden subgroup problem – this paper represents a departure from that tradition, and in doing so provides polynomial-time algorithms for several other important problems, including testing membership, subgroup equality, and subgroup normality within solvable groups. Watrous' algorithm also has the desirable side effect of producing a pure quantum state that is a uniform mixture of the elements in any specific subgroup of the group, which provides a natural and efficient way to apply existing group algorithms to factor groups.

1 Introduction

Group theory has, over the past century, become an increasingly important and elegant mathematical tool for describing and solving many problems within the sciences. Symmetry groups in particular have been used by chemists to simplify problems regarding molecular properties by viewing large, complex molecules in terms of their symmetry [6]. Within physics, Emmy Noether's theorem creates a link between the conservation laws of physics and mathematical groups, stating that every continuous symmetry corresponds to a conserved property; this observation leads to natural applications of Lie groups to physical systems [7]. Combined with its applications to cryptography, topology, and almost all other areas of mathematics, group theory is a powerful tool for understanding and solving larger problems.

Given the importance of the order of a group[1], it would be useful to have an efficient method for producing the order of an arbitrary group. Since no efficient classical algorithm exists, even if we can be assured that the group is abelian, Watrous' algorithm, while limited to solvable groups, provides a means to compute the order of many important groups efficiently. As an important side effect, the algorithm also allows computation over factor groups, another helpful tool in simplifying complex systems.

Previous important quantum algorithms mostly focus on the abelian hidden subgroup problem. This tradition began with Shor's 1994 algorithm for factorizing integers in polynomial time. While the algorithm was not explicitly stated in terms of black-box groups, it is easily generalized to a

[1]One of the main results of finite group theory, Lagrange's Theorem, states that for any subgroup H of a finite group G, the order of H is a divisor of G. This result has many important consequences, and is extremely helpful in understanding the structure of a group (its elements, subgroups, morphic images, etc.)

finite group framework. The original algorithm requires that the inputs (a and N) be relatively prime, as otherwise a need not have an order – within a group framework, we are assured that every element has such an order.

The integer factorization algorithm (specifically finding orders), as well as other early quantum algorithms like Simon's algorithm, have since been generalized as solving specific applications of the hidden subgroup problem:

> Let $f : G \to X$ map some group G to some finite set X, with the property that there exists a **hidden subgroup** S whereby $f(x) = f(y)$ if and only if $xS = yS$. Determine S

Many problems can be formalized in terms of hidden subgroups, and so much effort has been devoted to finding a general, efficient solution for it. There has been some success, as a generalized polynomial-time algorithm has been developed for the case when G is an abelian group [2], and with it many other problems have quantum polynomial-time algorithms. There have also been attempts to generate a polynomial-time algorithm for the general non-abelian case; however, there has been limited success in this area and it is generally believed that no efficient quantum algorithm for the non-abelian hidden subgroup exists. Some specific successes include proof of an efficient query complexity in the general case by M. Ettinger, P. Hoyer, and E. Knill [4], while G. Kuperberg has a sub-exponential quantum algorithm that solves the hidden subgroup problem for dihedral groups[5].

By always working in one specific problem space, the possibilities and techniques of quantum computing are not being explored as well as they could be. In the paper *Quantum Algorithms for Solvable Groups*, group theoretic problems outside the scope of the hidden subgroup problem are examined, an important step for quantum algorithms. While the paper presents multiple algorithms, the base result is an efficient[2] algorithm for computing the order of a solvable group G (and preparing a uniform superposition of G), effectively proving the following as stated in [1]:

Theorem 1. *There exists a quantum algorithm operating as follows (relative to an arbitrary group oracle). Given generators $g_1, ..., g_k$ such that $G = \langle g_1, ..., g_k \rangle$ is solvable, the algorithm outputs the order of G with probability of error bounded by ϵ in time polynomial in $n+log(1/\epsilon)$ (where n is the length of the strings representing the generators). Moreover, the algorithm produces a quantum state ρ that approximates the pure state $|G\rangle = |G|^{(} - 1/2) \sum_{g \in G} |g\rangle$ with accuracy ϵ (in the trace norm metric).*

The paper will follow largely in the same manner as [1], with extra clarity added at points and high-level descriptions aimed to explain the concepts behind the technical details. For the main algorithm, the mathematical details will also be thoroughly sketched out, and will follow a slightly different analysis using an eigenvector basis. This is done both to improve the accuracy of the analysis, and the clarity of understanding for those more acclimated to the eigenvalue estimate approach to order finding. Section 2 will detail some group theory background as well as black box groups, section 3 will describe in detail the main algorithm, and section 4 will consider the applications proposed and identify some potential issues not detailed in [1]. Finally, section 5 will serve to conclude the paper and consider the implications on both industry and complexity theory.

[2] Polynomial-time.

2 Preliminaries

Before examining the order finding algorithm, we will review some basic concepts necessary for its understanding. We will assume that readers are familiar with introductory quantum computation, as seen in *An Introduction to Quantum Computing* [2]. Additionally, we will assume that the reader has a cursory knowledge of group theory, though for the sake of accessibility we provide some basic group theoretic definitions. The group-theoretic definitions used throughout this paper include, though is not limited to, groups, (normal) subgroups, and generators.

Definition 2. (Group) A group G is a set together with a binary operation that assigns each ordered pair (a, b) of elements in G an element, denoted ab, in G, and satisfies the following properties:

1. **Associativity.** $(ab)c = a(bc)$ for all $a, b, c \in G$.
2. **Identity.** There exists an element $e \in G$ such that $ae = a = ea$ for all $a \in G$.
3. **Inverses.** For all $a \in G$, there exists an element $b \in G$ such that $ab = e = ba$.

(Subgroup) A subgroup of a group G is a subset $H \subseteq G$ that is closed under the group operation of G. A subgroup H of G is **normal** (denoted $H \lhd G$) if for all $a \in G, aH = Ha$.

The order of a group is the number of elements it contains, denoted $|G|$. It's worth noting that a **generating** set for a group G fulfills a similar role as a basis (for a vector space) does in linear algebra; while this is not an entirely accurate definition, it should be sufficient for readers unfamiliar with group theory to gain a general understanding of their usage.

2.1 The black-box model

As do many other results regarding quantum computation of group-theoretic problems, the algorithm described in [1] works within the context of black-box groups. There are many different formulations of black-box groups (see appendix 6 in [2] for a discussion of specific models and implementation issues), so we will just review the basic structure of a black-box group, as well as results that relate to the order-finding algorithm.

A black-box group consists of a group in which each element is encoded by a unique binary string, and an oracle that performs the computation of the group operation. A specific black-box group has an encoding length n, which denotes the (constant) number of bits used to encode any element within the group – so a black-box group with encoding length n can uniquely encode at most 2^n elements; though not all 2^n strings necessarily encode group elements. As an important note, by this definition all black-box groups are finite groups, so we can assume we are working with a finite group. Black-box groups are typically described by a set of strings that generate the group, ie. strings encoding a generating set. This assumption regarding black-box groups will be particularly important, as the algorithm requires knowledge of a generating set – however, there is little restriction on the size of the generating set.

The most general group oracle for a black-box group G is the unitary quantum gate U_G that implements the map $|g\rangle|h\rangle \to |g\rangle|gh\rangle$. Depending on the group and the operation, different oracles may be easier or more difficult to implement, but we assume U_G is available as it is the most general and convenient form. Interestingly, it is noted in [2] that an oracle implementing the group

operation as a bitwise exclusive-or on an ancilla cannot efficiently be used to multiply by the inverse of a group element, yet it is noted in [1] that the order of a group element can be found using Shor's algorithm and then used to find its inverse. While complexity theory would label this as an efficient method, it may not be plausible in practice to use Shor's algorithm to implement U_G^{-1}.

2.2 Solvable black-box groups

We now define what it means for a group to be solvable. While this is not the simplest definition, it is equivalent to a more basic definition as in [3].

Definition 3. (Solvable group) A group G is called *solvable* if there exist $g_1, ..., g_m \in G$ such that when we let $H_i = \langle g_1, ..., g_i \rangle$, we have $\{e\} = H_0 \lhd H_1 \lhd ... \lhd H_m = G$.

With this definition, the forementioned property that every abelian group is solvable does not immediately arise. It is therefore useful, just to convince us of the solvable property of abelian groups, to know that an equivalent definition does away with the requirement that $H_i = \langle g_1, ..., g_i \rangle$, and instead replaces it with the restriction that H_j/H_{j-1} is abelian. Since $\{e\} \lhd G$ in any group, if G is abelian then so is $G/\{e\}$, ie. G is solvable. As in interesting note, it can also be seen that every group of odd order is solvable [3].

As noted in [1], there are many known facts about black-box solvable groups that will be used implicitly in the order finding algorithm. In order to be able to use the structure of the chain shown in the definition of a solvable group, we need a theorem that ensures such a generating set can be found efficiently.

Theorem 4. *Given a group oracle and generators $g_1, ...g_m$ there exists a polynomial-time Monte Carlo algorithm for testing whether $G = \langle g_1, ..., g_m \rangle$ is solvable, as well as constructing generators $h_1, ..., h_k n$ such that defining $H_i = \langle h_1, ..., h_i \rangle$ gives $\{e\} = H_0 \lhd H_1 \lhd ... \lhd H_k n = G$, with $k \in O(n)$*

Though the specific details of the above result won't be described, the idea of the theorem is to build the derived subgroups[3] of G and use their generators to define the H_j's. Testing solvability is then achieved by verifying that the generators of the nth derived subgroup are all the identity.

3 Finding orders of solvable groups

This section will detail the algorithm for computing the order of a solvable black-box group G and preparing a uniform superposition over the elements of G, as described in *Quantum Algorithms for Solvable Groups*[1]. It will, for the most part, follow the analysis given in the paper and provide extra clarity where possible.

The algorithm hinges on an observation from group theory; by the solvability of G, we know there exist elements $g_1, ..., g_m \in G$ such that if $H_j = \langle g_1, ..., g_j \rangle$, these H_j's form an ascending chain of normal subgroups with $H_0 = \{e\}, H_m = G$. Then if we let $r_j = |H_j/H_{j-1}|$ for each $1 \leq j \leq m$, we know from basic group theory that $r_j = \frac{|H_j|}{|H_{j-i}|}$. Thus

$$\prod_{j=1}^{m} r_j = \frac{|H_1||H_2|...|H_m|}{|H_0||H_1|...|H_{m-1}|} = \frac{|H_m|}{|H_0|} = |G| \tag{1}$$

[3]See either [1] or [3] for a definition – it is not crucial to an understanding of the paper at this point.

With this observation, it's clear that if we can calculate the order of each factor group H_j/H_{j-1} efficiently, then we have an efficient algorithm to compute the order of G. We note that if we let r be the smallest positive r such that $g^r \in H_{j-1}$, called the *order of g_j with respect to H_{j-1}*, then $r = |H_j/H_{j-1}|$[4]. So we have now reduced finding the order of the group G to finding relative orders of its generating set. Since we know how to find the order of a group element efficiently using Shor's algorithm, we should be able to find this relative order efficiently with some modifications.

Of course, the order finding algorithm requires an initial identity state, which in this case is the coset H_j, so we also need a way to efficiently prepare copies of each subgroup H_j. It should be noted at this point that the algorithm given for the construction of such superpositions necessitates the condition that G is solvable. The basic idea is to use the order of g_j with respect to H_{j-1} to convert uniform superpositions of H_{j-1} to H_j. So at each step, the algorithm will need some copies of a uniform superposition of H_{j-1} to compute $r_j = |H_j/H_{j-1}|$, which are destroyed in the process due to measurement, and then additional copies to convert into uniform superpositions of H_j. While this requires a large number of initial registers in uniform superpositions of H_0 (which is easy to prepare, given that $H_0 = \{e\}$), it is still polynomial in the number of generating elements for any desired error probability.

As in the paper, we will sketch separate algorithms for both main components – finding orders with respect to a subgroup, and preparing uniform superpositions of a subgroup – and then describe the combined algorithm and analyze its complexity. We will denote the order of $g \in G$ with respect to subgroup H as $\boldsymbol{r_H(g)}$, and let the uniform superposition of a finite set S be denoted by $\boldsymbol{|S\rangle = \frac{1}{\sqrt{|S|}} \sum_{g \in S} |g\rangle}$

3.1 Finding orders with respect to a subgroup

Computing the order of g with respect to H can be accomplished by Shor's approach to order finding, replacing the initial state $\sum_{x=0}^{2^n-1} |x\rangle|1\rangle$ with $\sum_{x=0}^{2^n-1} |x\rangle|H\rangle$ and then applying the reversible map $|x\rangle|y\rangle \rightarrow |x\rangle|g^x y\rangle$. It will be seen that if we begin with the above states, then follow the remainder of the algorithm essentially verbatim, the result is $r_H(g)$. This process will be used within the main algorithm to find the order of g_j with respect to H_{j-1} for each j.

The key observation behind this modification is that Shor's algorithm can be generalized as a method to compute the period r of some computable, periodic function f. In the case of order finding and integer factorization, $f(x) = a^x mod N$, so that the period r is the order of a; on the other hand, if we let $f(x) = g^x H$, the period r is the lowest positive integer such that $g^x H = H$ and thus $g^x \in H$. However, care must be taken as Shor's algorithm requires that the states $|f(x)\rangle, |f(y)\rangle$ are orthogonal if $r \nmid x - y$ – fortunately, $g^i H \cap g^j H = \{e\}$ (so $|g^i H\rangle$ and $|g^j H\rangle$ are orthogonal) whenever $g^i H \neq g^j H$.

While the paper uses Shor's algorithm to provide the method, and defers the details of the analysis to external sources, we'll sketch the method using the equivalent eigenvalue estimation approach to order finding [2], as the analysis appears cleaner. We assume we are working with a black-box group G having encoding length n, and let $N \in O(2^n)$[5] be determined later to achieve a desired error probability. Suppose H is some subgroup of G, $g \notin H$ be a group element, and

[4]Every element of H_j can be written as $g_j^i g'$ for some $g' \in H_{j-1}$, so the factor group H_j/H_{j-1} consists of the cosets $g_j^i H_{j-1}$, exactly r of which will be unique

[5]The exponential bound may appear troublesome, but it is known that the QFT_N can be implemented in time polynomial in $\log N$ [2], thus we still have a polynomial-time algorithm

$r = r_H(g)$. Also, we let U_g implement the map $|h\rangle \to |gh\rangle$ for any $g, h \in G$. This circuit can be implemented classically and efficiently via repeated squaring, and is reversible on the basis that g is a group element so it has an inverse.

Before we describe the actual algorithm, we notice that as in the eigenvalue estimation approach, if we define

$$|\mu_k\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |g^s H\rangle \tag{2}$$

$|\mu_k\rangle$ is an eigenvalue of U_g – ie. $U_g|\mu_k\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |g^{s+1} H\rangle = e^{2\pi i \frac{k}{r}} |\mu_k\rangle$. However, if we have a uniform superposition of such states $|\mu_k\rangle$, then we see

$$\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\mu_k\rangle = \frac{1}{r} \sum_{s=0}^{r-1} (\sum_{k=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |g^s H\rangle) = \frac{1}{r} \sum_{k=0}^{r-1} |g^0 H\rangle = |H\rangle \tag{3}$$

since $\sum_{k=0}^{r-1} e^{-2\pi i \frac{k}{r} s} |g^s H\rangle = 0$ if $s \neq 0$.

Finally we sketch the algorithm:

1. Begin by preparing the state $|0\rangle^{\otimes n} |H\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |0\rangle^{\otimes n} |\mu_k\rangle$
2. Apply QFT_N to the control register to compute the state $\frac{1}{\sqrt{rN}} \sum_{k=0}^{r-1} \sum_{x=0}^{N-1} |x\rangle |\mu_k\rangle$
3. Apply $c - U_g^x$ to the control and target registers: $\frac{1}{\sqrt{rN}} \sum_{k=0}^{r-1} \sum_{x=0}^{N-1} e^{2\pi i \frac{k}{r} x} |x\rangle |\mu_k\rangle$
4. Apply QFT_N^{-1} to the control register and measure (the control register) to obtain an estimate $\frac{b}{N}$ of $\frac{k}{r}$ for some random $0 \leq k < r$

The remainder follows essentially identically to the analysis in [2], so it will be described only briefly. With high probability, $\frac{b}{N}$ is a good estimate, and if we let $N = 2^{2n + O(log(1/\epsilon))}$, using the continued fraction method we find, with probability $1 - \epsilon$, relatively prime integers u, v such that $\frac{u}{v} = \frac{k}{r}$. We then repeat the algorithm $O(log(1/\epsilon))$ times and compute the least common multiple of the v's to obtain r with probability $1 - \epsilon$.

3.2 Building uniform superpositions over subgroups

Building a uniform superposition over a subgroup $\langle g \rangle H$ is achieved in this method effectively by applying the order finding algorithm given above (using $QFT_{r_H(g)}$ instead of QFT_N) on a large number of copies of $|H\rangle$. Then, we find one register that contains all the $(r_H(g))^{th}$ roots of unity, and use it correct the rest of the registers, discarding it afterwards. Sepcifically, the algorithm converts copies of $|H\rangle$ into copies of $|\langle g \rangle H\rangle$ for some g. We note that this requires H to be normal in $|\langle g \rangle H\rangle$, as the internal direct product $G = HK$ of two groups H, K may not be a group if either H or K is not normal in G [3]; the solvability of G arises from this requirement. In terms of the actual algorithm, this method is used to convert l copies of $|H_{j-1}\rangle$ into $l - 1$ copies of $|\langle g_j \rangle H_{j-1}\rangle$, which is equal to $|H_j\rangle$ given the definition $H_j = \langle g_1, ..., g_j \rangle$.

We now account the method, assuming we have calculated $r = r_H(g)$, and have l copies of $|H\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |\mu_k\rangle$ as defined in the section above[6]. First prepare l copies of the state $|0\rangle^{\otimes r}|H\rangle$. Then, similar to calculating the order $r_H(g)$, we apply QFT_r to the control register, followed by $c - U_g^x$ to both registers, then finally QFT_r^{-1}. This process results in the state

$$\frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} |k\rangle|\mu_k\rangle = \frac{1}{r} \sum_{s=0}^{r-1} \sum_{k=0}^{r-1} e^{-2\pi i \frac{ks}{r}} |k\rangle|g^s H\rangle \tag{4}$$

Next, measure the control registers of all l copies, denoting the results $k_1, ..., k_l$, and labeling the resulting states $|\psi_i\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} e^{-2\pi i \frac{k_i s}{r}} |g^s H\rangle$. We pick one value j such that k_j is relatively prime to r. We can assure at least one such k_j exists with probability at least $1 - \epsilon$ by choosing $l \in \Omega$ $((\log \log r)(\log \frac{1}{\epsilon}))$[7].

The last part of the algorithm applies the insight that $|\psi_j\rangle$ is an eigenvector of the map $M_{g^x h} : |\omega\rangle \to |g^x h \omega\rangle$. Since $|\psi_j\rangle$ is in a superposition of all elements of $\langle g \rangle H$, multiplication by $g^x h$ simply shuffles their amplitudes. Specifically, if $s' = s + x \mod r$, then $s + mn = s' - x$ for some integer m, so

$$e^{2\pi i \frac{sk_j}{r}} |g^{s'} H\rangle = e^{2\pi i \frac{(s'-x)k_j}{r}} |g^{s'} H\rangle = e^{2\pi i \frac{-xk_j}{r}} e^{2\pi i \frac{s'k_j}{r}} |g^{s'} H\rangle \tag{5}$$

ie. $M_{g^x h}|\psi_j\rangle = e^{2\pi i \frac{-xk_j}{r}} |\psi_j\rangle$. So, for each $i \neq j$ let c be some integer satisfying $c \equiv k_i k_j \mod r$, and for each state $|g^s h\rangle$ in the first register, apply $M_{(g^s h)^c}$ to the second register in $|\psi_i\rangle|\psi_j\rangle$. By the above observation, this results in the state

$$\frac{1}{\sqrt{r|H|}} \sum_{s=0}^{r-1} \sum_{h \in H} e^{2\pi i \frac{sk_i - sck_j}{r}} |g^s h\rangle|\psi_j\rangle = \frac{1}{\sqrt{r|H|}} \sum_{s=0}^{r-1} \sum_{h \in H} e^{2\pi i \frac{sk_i - sk_i - smr}{r}} |g^s h\rangle|\psi_j\rangle = |\langle g \rangle H\rangle|\psi_j\rangle \tag{6}$$

So we have converted the $l - 1$ states $|\psi_i\rangle, i \neq j$ to the state $|\langle g \rangle H\rangle$ as required

3.3 Completing the algorithm

Now that we have separate (polynomial-time) algorithms for both (i) computing the order $r_H(g_j)$ of g_j with respect to H_{j-i}, and (ii) converting l copies of $|H_{j-i}\rangle$ to $l-1$ copies of $|\langle g_j \rangle H_{j-1}\rangle = |H_j\rangle$, we can sketch the composition of the entire algorithm. Recall that $H_0 = e, H_m = G$

1. Begin with $k(m+1)$ copies of $|H_0\rangle$ where $k \in O((\log n)(\log \frac{m}{\epsilon}))$ for a desired error probability, ϵ. Set j:=1
2. Using $k - 1$ copies of $|H_{j-1}\rangle$, compute $r_j = r_{H_{j-1}}(g_j)$
3. Convert the remaining copies of $|H_{j-1}\rangle$ to $|H_j\rangle$, losing one copy in the process to leave $k(m+1) - jk$ copies left
4. if $j = m$, output $\prod_{j=1}^{m} r_j$; else set $j := j + 1$ and return to (2).

[6]For the sake of continuity, the analysis follows in the eigenvalue basis, even though we will end up with the same state that arises from Shor's analysis and described in the paper, up to relabeling.

[7]This probability, as given in [1], was without proof or citation. However, it is noted that a modified version of the algorithm does not require there to be a relatively prime k; it would likely be helpful in practice to pursue such an algorithm, so as to minimize the error probability in exchange for more complex computations.

As detailed in [1], by choosing $k \in O((\log n)(\log \frac{m}{\epsilon}))$, the algorithm errs with probability at most ϵ. Using $O(log(2m/\epsilon))$ copies of $|H_{j-1}\rangle$ computes r_j with error probability at most $\epsilon/2m$, and since $O(log(2m/\epsilon))$, the total error in computing all m r_j's is at most $m\frac{\epsilon}{2m} = \epsilon/2$. Similarly, since $n \in O(\log r)$, we see that $k \in O((\log \log r)(\log \frac{1}{\epsilon/2m}))$ – since for each j, at least k states are being converted, the total error in converting the states is at most $m\frac{\epsilon}{2m} = \epsilon/2$. So the total error is at most $\epsilon/2 + \epsilon/2 = \epsilon$.

Watrous has therefore detailed a polynomial-time algorithm for computing the order of a solvable group G and creating a uniform superposition over the elements of G.

4 Applications

The general and unique nature of this algorithm allows a number of other group-theoretic problems to be solved efficiently (for solvable groups). While the advantages of knowing the order of a group are well known [3], the algorithm also allows computations over factor groups, an important mathematical tool for simplifying complex problems. However, these algorithms are not formally detailed, and as such there are issues that could arise if one were to formalize them.

4.1 Applications of order finding

Membership testing in a black-box solvable group reduces very naturally, through simple group theory, to solving the order of a group. If we consider the generating set $\{g_i\}$ of some black-box solvable group G and add another element h to its generating set (roughly equivalent to creating the internal direct product $\langle h \rangle G$), the resulting group will (a.) have the same order if $h \in G$, or (b.) strictly increase in order otherwise. So by computing the order of both groups and comparing the two, we have computed in polynomial time with certainty whether $h \in G$. Important to note is that if the group generated by $\{g_i\}$ and h is not solvable, as can be computed in polynomial time, then we instantly know $h \notin G$.

While this is a useful theoretical result, the formulation of a black-box group given earlier raises question about its implementation. Previously, we were only concerned with computation on elements of a black-box group G, and so the behaviour of the group oracle on an invalid encoding was left undefined; by running the algorithm on a possibly invalid encoding h, the algorithm is not necessarily well-defined. While an error due to an invalid encoding would, in most cases, likely be easy to detect (for example, if an invalid encoding causes the oracle to produce random states, it's unlikely that an algorithm relying on specific states would complete correctly), the behaviour of a specific black-box on invalid encodings still needs to be defined and accounted for to properly prove a polynomial-time algorithm. It is, however, not the main focus of the paper, and it is successful in sketching out a possible method for membership testing.

Other problems reducible to order finding (and membership testing) in solvable groups include subgroup testing, normality testing, and group equality (ie. equality of generating sets). They all involve computing the orders of different generating sets and comparing them, based on results from group theory. There are likely many more such problems that could be solved through order finding and membership testing, as many proofs and exercises in group theory are solved using group orders.

This algorithm also suggests succinct classical certificates for verifying that a given integer d

divides the order of a black-box solvable group G. A possible certificate would be composed of the p-subgroups of G for all prime divisors p of d, and then verification would proceed by testing whether each p-subgroup is in fact a subgroup of G. However, this particular certificate requires that G is solvable for its use of the order-finding algorithm; an assumption that trivializes the problem in any case, as we have a polynomial-time quantum algorithm for computing and factorizing $|G|$ (using Shor's algorithm to factorize $|G|$). Watrous has also previously shown that there exists a succinct quantum certificate for the same problem in his paper *Succinct quantum proofs for properties of finite groups.*

4.2 Applications of uniform superpositions over subgroups

As mentioned earlier, an important consequence of the order finding algorithm is that it provides a means for building uniform quantum superpositions of specific subgroups. These uniform superpositions can then be used to conveniently express elements of factor groups, which in the case of a solvable group will be abelian (since the subgroup the is built necessarily exists in a chain as per the definition of a solvable group). Since we can use the same group oracle to perform the coset multiplication operation, standard efficient techniques for computing properties of abelian groups, of which there are many, can be applied to this factor group.

The specific example given determines the classification of a factor group G/H, according to the finite theorem of abelian groups [3]. A high-level sketch of the method follows: assuming we have prepared the state $|H\rangle$ (using the order finding algorithm), and computed the LCM N of the orders of the generating set $g_1, ..., g_k$ of G (using Shor's algorithm), we can prepare the superposition of all $|a_1...a_k\rangle|g_1^{a_1}...g_k^{a_k}H\rangle$ for $0 \leq a_i < N$. By applying QFT_N on each of the first register k registers, we can then cancel the amplitudes so that measuring them yields $b_1...b_k$ where $g_1^{a_1}...g_k^{a_k} \in H$.

This particular problem was given as an example of the possible use of the ability to generate a uniform superposition over a subgroup, though many more possible uses exist. The approach has since been applied by G. Ivanyos, F. Magniez, and M. Santha to perform a constructive membership test in G using a normal, solvable subgroup H, and to find Sylow subgroups of G [11].

Kapovich, Myansnikov, Schupp, and Shpilrain since proven that certain black-box group problems, specifically membership within subgroups and conjugacy (and thus subgroup normality), are classically in P [10]. In addition, within specific domains such as networks,

5 Conclusion

So we have now detailed the polynomial-time quantum algorithm, formulated in *Quantum Algorithms for Solvable Groups*, for computing the order of a solvable group G, and preparing a uniform superposition over G. We have also examined the claims of the problems reducible to this algorithm, and identified a few possible issues with them.

One natural question this paper raises, is whether these methods can be adapted to non-solvable groups. The solvability of G is not strongly enforced in the algorithm, and it is plausible that a method of building uniform subgroups could be determined that does not rely on the normality assumption. Care must be taken, however, as a surgical removal of the solvable assumption may break other implicit assumptions made in the algorithm. The paper itself is concluded by suggesting that if generators of the Sylow subgroups of G could be computed, the order finding algorithm could be run on these (solvable) subgroups; it is not known to the author whether any papers taking this

approach have yet been published.

Since this paper was published in 2001, there have been further advancements in algorithms for black-box groups. One interesting results, proven in 2004 by Kapovich, Myansnikov, Schupp, and Shpilrain, is that certain black-box group problems, including membership within subgroups and conjugacy (and thus subgroup normality), are classically in P for a large class of groups (potentially larger than solvable groups)[10]. In this way, the proposed applications of the main algorithm appear somewhat less helpful in hindsight. In addition, we can observe that within many specific domains and industrial applications, such as permutation groups representing computer networks, there exist efficient heuristic algorithms making use of theorems like the stabilizer-orbit theorem to determine group orders [8].

Yet, Watrous' algorithm still represents an important step towards finding efficient algorithms for general group theoretic problems. Given that the order of a solvable group is hard to compute classically, it serves as further evidence that the a quantum computation model is strictly more powerful than a classical computation model.

References

[1] J. Watrous, *Quantum Algorithms for Solvable Groups*. http://arxiv.org/abs/quant-ph/0011023 (2001).

[2] P. Kaye, R. Laflamme, and M. Mosca, *An Introduction to Quantum Computing*. Oxford University Press (2007).

[3] J. Gallian, *Contemporary Abstract Algebra (seventh ed.)*. Brooks/Cole, Cengage Learning (2006).

[4] M. Ettinger, P. Hoyer, and E. Knill, *The quantum query complexity of the hidden subgroup problem is polynomial*. http://xxx.lanl.gov/abs/quant-ph/0401083 (2004).

[5] G. Kuperberg, *A subexponential-time quantum algorithm for the dihedral hidden subgroup problem*. http://xxx.lanl.gov/abs/quant-ph/0302112 (2004).

[6] C. Johnson, *Group Theory in Chemistry*. http://www.wpi.edu/Pubs/E-project/Available/E-project-042308-104731/unrestricted/ChaseJohnsonMQP.pdf (2008).

[7] N. Byers, *E. Noether's Discovery of the Deep Connection Between Symmetries and Conservation Laws*. http://www.physics.ucla.edu/ cwp/articles/noether.asg/noether.html.

[8] S. Stoichev, *Polynomial time and space exact and heuristic algorithms for determining the generators, orbits and order of the graph automorphism group*. http://arxiv.org/ftp/arxiv/papers/1007/1007.1726.pdf.

[9] L. Babai, *Computational Complexity in Finite Groups*. http://www.mathunion.org/ICM/ICM1990.2/Main/icm (1990).

[10] I. Kapovich, A. Myasnikov, P Schupp, and V. Shpilrain, *GENERIC-CASE COMPLEXITY, DECISION PROBLEMS IN GROUP THEORY AND RANDOM WALKS*. http://www.math.mcgill.ca/ alexeim/Publications/All_files_new/generic_last_55.pdf (2004).

[11] G. Ivanyos, F. Magniez, and M. Santha, *Efficient quantum algorithms for some instances of the non-abelian hidden subgroup problem.* http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.6936&rep=rep1&type=pdf.