SIAM J. COMPUT.
Vol. 35, No. 1, pp. 170–188

A SUBEXPONENTIAL-TIME QUANTUM ALGORITHM FOR THE DIHEDRAL HIDDEN SUBGROUP PROBLEM*

GREG KUPERBERG†

Abstract. We present a quantum algorithm for the dihedral hidden subgroup problem (DHSP) with time and query complexity $2^{O(\sqrt{\log N})}$. In this problem an oracle computes a function f on the dihedral group D_N which is invariant under a hidden reflection in D_N. By contrast, the classical query complexity of DHSP is $O(\sqrt{N})$. The algorithm also applies to the hidden shift problem for an arbitrary finitely generated abelian group.

The algorithm begins as usual with a quantum character transform, which in the case of D_N is essentially the abelian quantum Fourier transform. This yields the name of a group representation of D_N, which is not by itself useful, and a state in the representation, which is a valuable but indecipherable qubit. The algorithm proceeds by repeatedly pairing two unfavorable qubits to make a new qubit in a more favorable representation of D_N. Once the algorithm obtains certain target representations, direct measurements reveal the hidden subgroup.

Key words. quantum algorithm, dihedral hidden subgroup

AMS subject classifications. 81P, 68W

DOI. 10.1137/S0097539703436345

1. Introduction.

The hidden subgroup problem (HSP) in quantum computation takes as input a group G, a finite set S, and a black-box function (or oracle) $f : G \to S$. By promise there is a subgroup $H \subseteq G$ such that $f(a) = f(b)$ if and only if a and b are in the same (right) coset of H. The problem is to identify the subgroup H. We assume that G is given explicitly; black-box groups are a separate topic [13].

Shor's algorithm [22] solves HSP when $G = \mathbb{Z}$ in polynomial time in the length of the output. An important predecessor is Simon's algorithm [23] for the case $G = (\mathbb{Z}/2)^n$. Shor's algorithm extends to the general abelian case [14], to the case when H is normal [10], and to the case when H has few conjugates [9]. Since the main step in the generalized algorithm is the quantum character transform on the group algebra $\mathbb{C}[G]$, we will call it the *character algorithm*.

In the dihedral hidden subgroup problem (DHSP), G is the dihedral group D_N and H is generated by a reflection. (Other subgroups of D_N are only easier to find; see Proposition 2.1.) In this case H has many conjugates and the character algorithm works poorly. This hidden subgroup problem was first considered by Ettinger and Høyer [7]. They presented an algorithm that finds H with a linear number of queries (in the length of the output) but an exponential amount of computation. Ettinger, Høyer, and Knill generalized this result to the general finite hidden subgroup problem [8].

In this paper we will describe a new quantum algorithm for the dihedral group D_N with a favorable compromise between query complexity and computation time per query.

*Received by the editors October 21, 2003; accepted for publication (in revised form) January 5, 2005; published electronically October 3, 2005. This research was supported by NSF grant DMS 0072342.

http://www.siam.org/journals/sicomp/35-1/43634.html

†Department of Mathematics, University of California, Davis, CA 95616 (greg@math.ucdavis.edu).

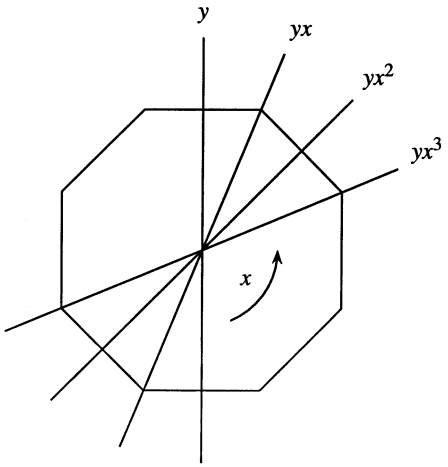

FIG. 1. *Some elements of* D_8.

THEOREM 1.1. *There is a quantum algorithm that finds a hidden reflection in the dihedral group* $G = D_N$ *(of order* $2N$*) with time and query complexity* $2^{O(\sqrt{\log\ N})}$.

The time complexity $2^{O(\sqrt{\log\ N})}$ is not polynomial, but it is subexponential. By contrast any classical algorithm requires at least $2N^{1/2}$ queries on average. Unfortunately, our algorithm also requires $2^{O(\sqrt{\log\ N})}$ quantum space.

We will prove Theorem 1.1 in a convenient case, $N = 2^n$, in section 3. In section 5, we will provide another algorithm that works for all N, and we will obtain the sharper time and query complexity bound $\widetilde{O}(3^{\sqrt{2\log_3 N}})$ when $N = r^n$ for some fixed radix r. The algorithm for this last case generalizes to many other smooth values of N.

2. Group conventions. The dihedral group D_N with $2N$ elements has the conventional presentation

$$D_N = \langle x, y \mid x^N = y^2 = yxyx = 1 \rangle.$$

(See Artin [2, section 5.3].) An element of the form x^s is a *rotation* and an element of the form yx^s is a *reflection*. The parameter s is the *slope* of the reflection yx^s. This terminology is motivated by realizing D_N as the symmetry group of a regular N-gon in the plane (Figure 1). In this model yx^s is a reflection through a line which makes an angle of $\frac{\pi s}{N}$ with the reflection line of y.

In this paper we will describe algorithms for the hidden subgroup problem with $G = D_N$ and $H = \langle yx^s \rangle$. If we know that the hidden subgroup is a reflection, then the hidden subgroup problem amounts to finding its slope s.

PROPOSITION 2.1. *Finding an arbitrary hidden subgroup* H *of* D_N *reduces to finding the slope of a hidden reflection.*

Proof. If H is not a reflection, then either it is the trivial group or it has a nontrivial intersection with the cyclic subgroup $C_N = \langle x \rangle$. Finding the hidden subgroup $H' = H \cap C_N$ in C_N is easy if we know the factors of N, and we can factor N using Shor's algorithm. Then the quotient group H/H' is either trivial or a reflection in the quotient group G/H'.

If H is trivial, then this will be revealed by the fact that an algorithm to find the slope of a hidden reflection must fail. □

3. A basic algorithm. In this section we will describe an algorithm to find the slope s of a hidden reflection in D_N when the period $N = 2^n$ is a power of 2. The main part of the algorithm actually finds only the parity of s. Once this parity is known, the main part can be repeated with a subgroup of D_N isomorphic to $D_{N/2}$. The group D_N has two such subgroups:

$$F_0 = \langle x^2, y\rangle, \qquad F_1 = \langle x^2, yx\rangle.$$

The subgroup $F_{s \bmod 2}$ contains H and the other does not, so we can pass to one of these subgroups if and only if we know s mod 2.

For any finite set S, the notation $\mathbb{C}[S]$ denotes a Hilbert space with S as an orthogonal basis. (This is the quantum analogue of a classical data type that takes values in S.) Define the *constant pure state* $|S\rangle$ in $\mathbb{C}[S]$, or more generally in $\mathbb{C}[T]$ for any $T \supseteq S$, as the superposition

$$|S\rangle = \frac{1}{\sqrt{|S|}} \sum_{s \in S} |s\rangle.$$

For the moment let us assume an arbitrary finite hidden subgroup problem $f : G \to S$ with hidden subgroup H. Assuming that there is a classical circuit to compute f, we can dilate it to a unitary embedding

$$U_f : \mathbb{C}[G] \to \mathbb{C}[G] \otimes \mathbb{C}[S] = \mathbb{C}[G \times S]$$

which evaluates f in the standard basis:

$$U_f|g\rangle = |g, f(g)\rangle.$$

All finite hidden subgroup algorithms, including ours, begin by computing

$$U_f|G\rangle$$

and then discarding the output register $\mathbb{C}[S]$, leaving the input register for further computation. The result is the mixed state

$$\rho_{G/H} = \frac{1}{|G|} \sum |Ha\rangle\langle Ha|$$

on the input register $\mathbb{C}[G]$.

Many works on hidden subgroup algorithms describe these steps differently [22, 18, 7, 8, 9, 10]. Instead of defining U_f as an embedding that creates $f(g)$, they define it as a unitary operator that adds $f(g)$ to an ancilla. They describe its output as measured rather than discarded, and they describe the mixed state $\rho_{G/H}$ as a randomly chosen coset state $|Ha\rangle$. We have presented an equivalent description in the formalism of mixed states and quantum operations [18, Chapter 8].

Now let $G = D_N$ with $N = 2^n$. The general element of D_N is $g = y^t x^s$ with $s \in \mathbb{Z}/N$ and $t \in \mathbb{Z}/2$. Thus the input register $\mathbb{C}[D_N]$ consists of n qubits to describe s and 1 qubit to describe t. The second step of our algorithm is to apply a unitary operator to $\rho_{D_N/H}$ which is almost the character transform (section 8.2). Explicitly, we apply the quantum Fourier transform (QFT) to $|s\rangle$,

$$F_N : |s\rangle \mapsto \frac{1}{\sqrt{N}} \sum_k e^{2\pi i k s/N} |k\rangle,$$

and then measure $k \in \mathbb{Z}/N$. The measured value is uniformly random, while the state on the remaining qubit is

$$|\psi_k\rangle \propto |0\rangle + e^{2\pi iks/N}|1\rangle.$$

(The symbol $\propto$ means "proportional to," so that we can omit normalization and global phase.) We will always create the same state $\rho_{D_N/H}$ and perform the same measurement, so we can suppose that we have a supply of $2^{O(\sqrt{n})}$ states $|\psi_k\rangle$, each with its own known but random value of k.

Note that $|\psi_{-k}\rangle$ and $|\psi_k\rangle$ carry equivalent information about s, because

$$|\psi_{-k}\rangle = X|\psi_k\rangle, \tag{1}$$

where X is the bit flip operator. They will be equivalent in our algorithms as well.

We would like to create the state

$$|\psi_{2^{n-1}}\rangle \propto |0\rangle + (-1)^s|1\rangle$$

because its measurement in the $|\pm\rangle$ basis reveals the parity of s. To this end we create a sieve which creates new $|\psi_k\rangle$'s from pairs of old ones. The sieve increases the number of trailing zeroes $\alpha(k)$ in the binary expansion of k. Given $|\psi_k\rangle$ and $|\psi_\ell\rangle$, their joint state is

$$|\psi_k\rangle \otimes |\psi_\ell\rangle \propto |0,0\rangle + e^{2\pi iks/N}|1,0\rangle + e^{2\pi ik\ell/N}|0,1\rangle + e^{2\pi i(k+\ell)/N}|1,1\rangle.$$

We now apply a CNOT gate

$$|a,b\rangle \mapsto |a,a+b\rangle$$

and measure the right qubit. The left qubit has the residual state

$$|\psi_{k\pm\ell}\rangle \propto |0\rangle + e^{2\pi i(k\pm\ell)s/N}|1\rangle$$

and the label $k \pm \ell$, which is inferred from the measurement of $a+b$. Thus we have a procedure to extract a new qubit $|\psi_{k\pm\ell}\rangle$ from the old qubits $|\psi_k\rangle$ and $|\psi_\ell\rangle$. The extraction makes an unbiased random choice between $k+\ell$ and $k-\ell$. We may well like the extracted qubit better than either of the old ones.

By iterating qubit extraction, we can eventually create the state that we like best, $|\psi_{2^{n-1}}\rangle$. We will construct a sieve that begins with $2^{\Theta(\sqrt{n})}$ qubits. Each stage of the sieve will repeatedly find two qubits $|\psi_k\rangle$ and $|\psi_\ell\rangle$ such that k and ℓ agree in $\Theta(\sqrt{n})$ low bits in addition to their trailing zeroes. With probability $\frac{1}{2}$, the label $k \pm \ell$ of the extracted qubit has $\sqrt{n}$ more trailing zeroes than k or ℓ. If the sieve has depth $\Theta(\sqrt{n})$, we can expect it to produce copies of $|\psi_{2^{n-1}}\rangle$.

In conclusion, here is a complete description of the algorithm to find a hidden reflection in D_N with $N = 2^n$. Also let $m = \lceil\sqrt{n-1}\rceil$.

ALGORITHM 3.1. *Input: An oracle $f : D_N \to S$ with a hidden subgroup $H = \langle yx^s\rangle$ and $N = 2^n$.*

1. *Make a list L_0 of copies of the state $\rho_{D_N/H}$ by applying the dilation D_f to the constant pure state $|D_N\rangle$ and discarding the input. Extract $|\psi_k\rangle$ from each $\rho_{D_N/H}$ with a QFT-based measurement.*
2. *For each $0 \le j < m$, we assume a list L_j of qubit states $|\psi_k\rangle$ such that k has at least mj trailing zeroes. Divide L_j into pairs of qubits $|\psi_k\rangle$ and $|\psi_\ell\rangle$ that share at least m low bits (in addition to trailing zeroes), or $n-1-mj$ bits if $m = j-1$. Extract the state $|\psi_{k\pm\ell}\rangle$ from each pair. Let the new list L_{j+1} consist of those qubit states of the form $|\psi_{k-\ell}\rangle$.*

3. *The final list L_m consists of states $|\psi_0\rangle$ and $|\psi_{2^{n-1}}\rangle$. Measure a state $|\psi_{2^{n-1}}\rangle$ in the $|\pm\rangle$ basis to determine the parity of the slope s.*
4. *Repeat steps* 1–3 *with the subgroup of D_N which is isomorphic to $D_{N/2}$ and which contains H.*

3.1. Proof of the complexity.

THEOREM 3.2. *Algorithm* 3.1 *requires $O(8^{\sqrt{n}})$ queries and $\widetilde{O}(8^{\sqrt{n}})$ computation time.*

Proof. In outline, if $|L_j| \gg 2^m$, then we can pair almost all the elements of L_j so that k and ℓ share m low bits for each pair $|\psi_k\rangle$ and $|\psi_\ell\rangle$. Then about half the pairs will form L_{j+1}, so that

$$\frac{|L_{j+1}|}{|L_j|} \approx \frac{1}{4}.$$

We can set $|L_m| = \Theta(2^m)$. Working backward, we can set $|L_0| = \Theta(8^m)$. The computation time consists of tasks with only logarithmic overhead.

In detail, we will assume that

$$|L_j| \geq C_{m-j} 2^{3m-2j}$$

for a certain constant $9 > C_k \geq 3$. We will bound the probability that this assumption survives as j increases. The constants are defined by letting $C_0 = 3$ and letting

$$C_k = \frac{C_{k-1}}{1 - 2^{-k-\frac{m}{3}}} + 2^{-2k}$$

by induction on k. It is not hard to check that

$$C_k > C_{k-1}, \qquad \lim_{k\to\infty} C_k < 9.$$

(A calculator may help for the first few terms of the limit, the worst case being $m = 1$.)

Since we create L_0 directly from oracle calls, we can set

$$|L_0| = C_0 2^{3m}.$$

Given L_j, let P_j be a maximal set of pairs $|\psi_k\rangle$ and $|\psi_\ell\rangle$ with m low matching bits. Then

$$|P_j| \geq \frac{|L_j| - 2^m}{2} \geq \frac{2^{3m-2j} C_j (1 - 2^{2j-2m})}{2},$$

because there are at most 2^m unmatched pairs. The list L_{j+1} is then formed from P_j by summand extraction, so $|L_{j+1}|$ can be understood as the sum of N independent, unbiased Bernoulli random variables. In general, if B_N is a sum of N unbiased Bernoulli random variables, then

$$P[B_N \leq \frac{(1-b)N}{2}] \leq (\cosh b)^N e^{-Nb^2} \leq e^{-Nb^2/2}.$$

(The first inequality is the Chernoff bound on large deviations.) Setting

$$b = 2^{j-\frac{4m}{3}},$$

we learn that

$$|L_{j+1}| \geq \frac{2^{3m-2j}(C_j - 2^{2j-2m})(1 - 2^{j-\frac{4m}{3}})}{4} = C_{j+1}2^{3m-2j-2}$$

with probability at least

$$1 - e^{-2^{\frac{m}{3}-1}}.$$

Finally by induction on j,

$$P[|P_j| \geq C_{m-j}2^{3m-2j}\ \forall j] \geq (1 - e^{-2^{\frac{m}{3}-1}})^m \to 1$$

as $m \to \infty$.

Thus the final list L_m is very likely to be large. Since the highest bit of k in $|\psi_k\rangle$ was never used for any decisions in the algorithm, it is unbiased Bernoulli for each entry of L_m. Therefore L_m is very likely to contain copies of $|\psi_{2^{n-1}}\rangle$. □

4. Some motivation. Algorithm 3.1 can be motivated by related ideas in representation theory and the theory of classical algorithms.

On the representation theory side, the input space $\mathbb{C}[D_N]$ has an orthogonal decomposition into two-dimensional representations V_k of D_N,

$$\mathbb{C}[D_N] \cong \bigoplus_{k \in \mathbb{Z}/N} V_k. \tag{2}$$

This means that each element of D_N is represented by a unitary operator on $\mathbb{C}[D_N]$ (given by left multiplication) and each V_k is an invariant subspace, so that each element of D_N is also represented by a unitary operator on each V_k [2, section 9.2]. Every orthogonal decomposition of a Hilbert space corresponds to a projective measurement [18, section 2.2.5]; this particular measurement can be computed using a QFT.

In the representation V_k, the generators x and y are represented as follows:

$$x \mapsto \begin{pmatrix} e^{2\pi/N} & 0 \\ 0 & e^{-2\pi/N} \end{pmatrix}, \qquad y \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Since the state $|Ha\rangle$ is invariant under the represented action of H, the residual state $|\psi_k\rangle$ is too. Thus abstract representation theory motivates the use of this state to find H. Note also that $V_k \cong V_{-k}$ as representations, as if reflected in the equivalence between $|\psi_k\rangle$ and $|\psi_{-k}\rangle$ in (1).

The representation V_k is irreducible except when $k = 0$ or $k = N/2$. Thus (2) is not far from the Burnside decomposition of $\mathbb{C}[G]$ into irreducible representations in the special case $G = D_N$. When expressed as a unitary operator, the Burnside decomposition is called the *character transform* or the noncommutative Fourier transform. (Measuring the character name solves the hidden subgroup problem for normal subgroups [10] and almost normal subgroups [9].) Use of $V_{N/2}$ as the target of Algorithm 3.1 is motivated by its reducibility; the measurement corresponding to its irreducible decomposition is the one that reveals the slope of s.

On the algorithm side, the sieve in Algorithm 3.1 is similar to a sieve algorithm for a learning problem due to Blum, Kalai, and Wasserman [5] and to a sieve to find shortest vector in a lattice due to Ajtai, Kumar, and Sivakumar [1].

Ettinger and Høyer [7] observed that if the state $|\psi_k\rangle$ for the hidden subgroup $H = \langle x^s y\rangle$ will be found in the state $|\psi'_k\rangle$ for a reference subgroup $H' = \langle x^t y\rangle$ with probability

$$\cos(\pi i(s-t)k/N)^2.$$

Thus the state $|\psi_k\rangle$ can provide a coin flip with this bias. We call such a coin flip a *cosine observation* of the slope s. Ettinger and Høyer showed that s is revealed by a maximum likelihood test with respect to $O(\log\ N)$ cosine observations with random values of k. They suggested a brute-force search to solve this maximum likelihood problem. Our first version of Algorithm 3.1 was a slightly subexponential, classical sieve on cosine observations that even more closely resembles the Blum–Kalai–Wasserman algorithm. Replacing the cosine observations by the qubit states $|\psi_k\rangle$ themselves significantly accelerates the algorithm.

5. Other algorithms. Algorithm 3.1 presents a simplified sieve which is close to the author's original thinking. But it is neither optimal nor fully general. In this section we present several variations which are faster or more general.

The first task is to prove Theorem 1.1 when N is not a power of 2. Given any qubit state $|\psi_k\rangle$, we can assume that $0 \le k \le \frac{N}{2}$, since $|\psi_k\rangle$ and $|\psi_{-k}\rangle$ are equivalent. The list L_j will consist of qubits $|\psi_k\rangle$ with

$$0 \le k < 2^{m^2-mj+1},$$

where

$$m = \lceil \sqrt{(\log_2 N) - 2} \rceil.$$

Another difference when N is not a power of 2 is that the quantum Fourier transform on $\mathbb{Z}/N$ is more complicated. An efficient approximate algorithm was given by Kitaev [14]; another algorithm which is exact (in a sense) is due to Mosca and Zalka [17].

ALGORITHM 5.1. *Input: An oracle $f : D_N \to S$ with a hidden subgroup $H = \langle yx^s\rangle$.*

1. *Make a list L_0 of copies of $\rho_{D_N/H}$. Extract a qubit state $|\psi_k\rangle$ from each $\rho_{D_N/H}$ using a QFT on $\mathbb{Z}/N$ and a measurement.*
2. *For each $0 \le j < m$, we assume a list L_j of qubit states $|\psi_k\rangle$ such that $0 \le k \le 2^{m^2-mj+1}$. Randomly divide L_j into pairs of qubits $|\psi_k\rangle$ and $|\psi_\ell\rangle$ that such that*

$$|k-\ell| \le 2^{m^2-m(j+1)+1}.$$

   Let the new list L_{j+1} consist of those qubit states of the form $|\psi_{|k-\ell|}\rangle$.
3. *The final list L_m consists of states $|\psi_0\rangle$ and $|\psi_1\rangle$. Perform the Ettinger–Høyer measurement on the copies of $|\psi_1\rangle$ with different values of t to learn $s \in \mathbb{Z}/N$ to within $N/4$.*
4. *Write $N = 2^a M$ with M odd. By the Chinese remainder theorem,*

$$C_N \cong C_{2^a} \times C_M.$$

   For each $1 \le j \le \lceil \log_2 N \rceil$, apply Algorithm 3.1 to produce many $|\psi_k\rangle$ with $2^{\min(a,j)}|k$. Then repeat steps 1–4 after applying the group automorphism $x \mapsto x^{2^{-j}}$ to the C_M factor of D_N. This produces copies of $|\psi_{2^j}\rangle$, and hence cosine observations $\cos(\pi i 2^j(s-t)/N)^2$. These observations determine s.

TABLE 1
*Average canceled bits in a simulation (*100 *trials).*

Queries	3	3^2	3^3	3^4	3^5	3^6	3^7	3^8
Zeroed bits	3.62	6.75	12.53	19.07	27.14	36.44	47.51	59.76
$\sqrt{2\log_3 2^n}$	2.14	2.92	3.98	4.91	5.85	6.78	7.74	8.68

The proof of Theorem 3.2 carries-over to show that Algorithm 5.1 also requires only $O(8^{\sqrt{\log_2 N}})$ queries and quasi-linear time in its data. The only new step is to check that in the final list L_m, the qubit states $|\psi_0\rangle$ and $|\psi_1\rangle$ are almost equally likely. This is a bit tricky but inevitable, given that the lowest bit of k can be almost uncorrelated with the way that $|\psi_k\rangle$ is paired.

Remark 5.2. Høyer described a simplification of Algorithm 5.1 [11]. Given only one copy each of

$$|\psi_1\rangle, |\psi_2\rangle, \ldots, |\psi_{2^k}\rangle,$$

with $2^k \geq N$, the slope s can be recovered directly by a quantum Fourier transform. More precisely, the measured Fourier number t of these qubits reveals s by the relation

$$\frac{t}{2^k} \sim \frac{s}{N}.$$

This simplification saves a factor of $O(\log N)$ computation time.

Now suppose that $N = r^n$ for some small radix r; Algorithm 3.1 generalizes to this case with only slight changes. It is natural to accelerate it by recasting it as a greedy algorithm. To this end, we define an objective function $\alpha(k)$ that expresses how much we like a given state $|\psi_k\rangle$. Namely, let αk be the number of factors of r in k with the exception that $\alpha(0) = 0$. Within the list L of qubit states available at any given time, we will greedily pick $|\psi_k\rangle$ and $|\psi_\ell\rangle$ to maximize $\alpha(k \pm \ell)$. It is also natural to restrict our greed to the qubits that minimize α, because there is no advantage to postponing their use in the sieve.

ALGORITHM 5.3. *Input: An oracle $f : D_N \to S$ with a hidden subgroup $H = \langle yx^s \rangle$ and $N = r^n$.*

1. *Make a list L of qubit states $|\psi_k\rangle$ extracted from copies of $\rho_{D_N/H}$.*
2. *Within the sublist L' of L that minimizes α, repeatedly extract $|\psi_(\rangle k \pm \ell)$ from a pair of qubits $|\psi_k\rangle$ and $|\psi_\ell\rangle$ that maximize $\alpha(k \pm \ell)$.*
3. *After enough qubits $|\psi_k\rangle$ appear with $\frac{N}{r}|k$, measure $s \mod r$ using state tomography. Then repeat the algorithm with a subgroup of D_N isomorphic to $D_{N/r}$.*

The behavior of Algorithm 5.3 (but not its quantum state) can be simulated by a classical randomized algorithm. We include the source code of a simulator written in Python with this article [15] with $r = 2$. Our experiments with this simulator led to a false conjecture for algorithm's precise query complexity. Nonetheless we present some of its results in Table 1. The last line of the table is roughly consistent with Theorem 5.4. Note that the sieve is a bit more efficient when $r = 2$ because then $k \pm \ell$ increases by 1 in the unfavorable case and at least 2 in the favorable case.

THEOREM 5.4. *Algorithm* 5.3 *requires $\widetilde{O}(3^{\sqrt{2\log_3 N}})$ queries and quasi-linear time in the number of queries.*

Here is a heuristic justification of the query bound in Theorem 5.4. We assume, as the proof will, that $r = 3$ and $N = 3^n$. Then with $3^{\sqrt{2n}}$ queries, we can expect

qubit extraction to initially cancel about $\sqrt{2n}$ ternary digits (trits) with probability $\frac{1}{2}$. If we believe the query estimate for $n' < n$, then we can expect the new qubit to be about 3 times as valuable as the old one, since

$$\sqrt{2n} - \sqrt{2n - \sqrt{2n}} \approx 1.$$

Such a qubit extraction trades 2 qubits for 1 qubit, which is half the time equivalent to the original 2 and half the time 3 times as valuable. Thus each step of the sieve breaks even; it is like a gamble with \$2 that is equally likely to return \$1 or \$3.

Sketch of proof. We will show that the sieve produces states $|\psi_{aN/r}\rangle$ (which we will call *final states*) with adequate probability when provided with at least $Cn3^{\sqrt{2\log_3 N}}$ queries. The work per query is quasi-linear in $|L|$ (initially the number of queries) if the list L is dynamically sorted. To simplify the formulas, we assume that $r = 3$, although the proof works for all r.

We can think of a qubit state $|\psi_k\rangle$ as a monetary asset, valued by the function

$$V(k) = 3^{-\sqrt{2(n-1-\alpha(k))}}.$$

Thus the total value $V(L)$ of the initial list L is at least

$$V(L) \geq Cn.$$

We claim that over a period of the sieve that increases $\min \alpha$ by 1, the expected change in $V(L)$ is at worst $-C$. Since $\min \alpha$ can only increase $n-1$ times, $V(L) \geq C$ when $\min \alpha = n - 1$. Thus the sieve produces at least C final states on average. Along the way, the changes to $V(L)$ are independent (but not identically distributed) Bernoulli trials. One can show using a version of the Chernoff bound (as in the proof of Theorem 3.2) that the number of final states is not maldistributed. We will omit this refinement of the estimates and spell out the expected behavior of $V(L)$.

Given k, let

$$\beta = \beta(k) = n - 1 - \alpha(k)$$

for short, so that β can be thought of as the number of uncancelled trits in the label k of $|\psi_k\rangle$. Suppose that two labels k and ℓ or $-\ell$ share m trits in addition to $\alpha(k)$ cancelled trits. Then

$$V(k) = V(\ell) = 3^{-\sqrt{2\beta}}. \tag{3}$$

The state $|\psi_{k\pm\ell}\rangle$ extracted from $|\psi_k\rangle$ and $|\psi_\ell\rangle$ has the expected value

$$\begin{aligned} E[V(k \pm \ell)] &= \frac{3^{-\sqrt{2\beta}} + 3^{-\sqrt{2(\beta-m)}}}{2} \\ &> 2V(k)\frac{1 + 3^{m/\sqrt{2\beta}}}{4}, \end{aligned} \tag{4}$$

using the elementary relation

$$\sqrt{2\beta} - \sqrt{2(\beta - m)} = \frac{2m}{\sqrt{2\beta} + \sqrt{2(\beta - m)}} > \frac{m}{\sqrt{2\beta}}.$$

The most important feature of (4) is that if $m > \sqrt{2\beta}$, the expected change in $V(L)$ is positive. Thus in bounding the attrition of $V(L)$, we can assume that $m \leq \sqrt{2\beta}$ for the best-matching qubits $|\psi_k\rangle$ and $|\psi_\ell\rangle$ in the sublist L' that minimizes α. By the pigeonhole principle, this can happen only when

$$|L'| \leq 3^{\sqrt{2\beta}}.$$

(To apply the pigeonhole principle properly, use the equivalence between $|\psi_k\rangle$ and $|\psi_{-k}\rangle$ to assume that the first nonzero digit is 1. There are then 3^m choices for the next m digits.)

When qubit extraction decreases $V(L)$, it decreases by at worst the value of one parent, given by the right side of (3). Likewise, if $|L'| = 1$ and its unique element $|\psi_k\rangle$ must be discarded, the loss to $V(L)$ is again the right side of (3). Thus the total expected loss as L' is exhausted is at most

$$3^{-\sqrt{2\beta}} 3^{\sqrt{2\beta}} < 1.$$

We can therefore take $C = 1$, although a larger C may be convenient to facilitate the Chernoff bound. □

Remark 5.5. A close examination of Algorithm 5.3 and Theorem 5.4 reveals that the sieve works with the same complexity bound if N factors as

$$N = N_1 N_2 \dots N_m$$

and N_k is within a bounded factor of 3^k. In this case the sieve will determine s mod N_1. This is enough values of N to extend to an algorithm for all N by the method of spliced approximation section 7.

6. Generalized dihedral groups and hidden shifts. In this section we consider several other problems that are equivalent or closely related to the hidden dihedral subgroup problem.

In general if A is an abelian group, let $\exp(A)$ denote the multiplicative form of the same group. Let $C_n = \exp(\mathbb{Z}/n)$ be the multiplicative cyclic group of order n. If A is any abelian group, define the *generalized dihedral group* to be the semidirect product

$$D_A \cong C_2 \ltimes \exp(A)$$

with the conjugation relation

$$x^{-1} = yxy$$

for all $x \in \exp(A)$ and for the nontrivial $y \in C_2$. Any element of the form yx is a *reflection* in D_A.

Suppose that A is an abelian group and $f, g : A \to S$ are two injective functions that differ by a shift:

$$f(a) = g(a + s).$$

Then the task of finding s from f and g is the *abelian hidden shift problem.* Another problem is the hidden reflection problem in A (as opposed to in D_A). In this problem, $f : A \to S$ is a function which is injective except that

$$f(a) = f(s - a)$$

TABLE 2
An oracle that hides $\langle yx^3 \rangle$ in D_8 and its hidden shift.

a	1	x	x^2	x^3	x^4	x^5	x^6	x^7
$f(a)$	A	B	C	D	E	F	G	H
a	y	yx	yx^2	yx^3	yx^4	yx^5	yx^6	yx^7
$f(a)$	F	G	H	A	B	C	D	E

for some hidden s.

PROPOSITION 6.1. *If A is an abelian group, the hidden shift and hidden reflection problems in A are equivalent to the hidden reflection problem in D_A.*

See Table 2 for an example.

Proof. If $a \in A$, let x^a denote the corresponding element in $\exp(A)$. Given $f, g : A \to S$, define

$$h(x^a) = f(a), \qquad h(yx^a) = g(a).$$

Then evidently

$$h(x^a) = h(yx^{s+a})$$

if and only if

$$f(a) = g(a + s).$$

We can also reduce the pair f and g to a function with a hidden reflection. Namely, let $S^{(2)}$ be the set of unordered pairs of elements of S and define $h : A \to S^{(2)}$ by

$$h(a) = \{f(-a), g(a)\}.$$

Then h is injective save for the relation

$$h(a) = h(s - a).$$

Conversely, suppose that $h : A \to S$ is injective save for the relation

$$h(a) = h(s - a).$$

If there is a $v \in A$ such that $2v \neq 0$, define

$$f : A \to S^{\times 2}, \qquad g : A \to S^{\times 2}$$

by

$$f(a) = (h(-a), h(v - a)), \qquad g(a) = (h(a), h(a - v)).$$

(If A is cyclic, we can just take $v = 1$.) Then f and g are injective and

$$f(a) = g(a + s).$$

If all $v \in A$ satisfy $2v = 0$, then h hides a subgroup of A generated by s, so we can find s by Simon's algorithm. ∎

Note also that Proposition 2.1 generalizes readily to generalized dihedral subgroups: finding a hidden reflection in D_A is as difficult as finding any hidden subgroup.

A final variation of DHSP is the hidden substring problem. In the $N \hookrightarrow M$ hidden substring problem,

$$f : \{0, 1, 2, \ldots, N-1\} \to S,$$
$$g : \{0, 1, 2, \ldots, M-1\} \to S$$

are two injective functions such that f is a shifted restriction of g, i.e.,

$$f(x) = g(x + s)$$

for all $0 \leq x < N$ and for some fixed $0 \leq s < M - N$.

7. More algorithms. In this section we will establish a generalization of Theorem 1.1 and a corollary.

THEOREM 7.1. *The abelian hidden shift problem has an algorithm with time and query complexity* $2^{O(\sqrt{n})}$*, where* n *is the length of the output, uniformly for all finitely generated abelian groups.*

COROLLARY 7.2. *The* $N \hookrightarrow 2N$ *hidden substring problem has an algorithm with time and query complexity* $2^{O(\sqrt{\log\ N})}$.

The proof of Corollary 7.2 serves as a warm-up to the proof of Theorem 7.1. It introduces a technique for converting hidden shift algorithms that we call *spliced approximation*.

Proof of Corollary 7.2. Identify the domain of f with $\mathbb{Z}/N$ (no matter that this identification is artificial). Make a random estimate t for the value of s, and define $h : D_N \to S$ by

$$g'(n) = g(n + t).$$

If t is a good estimate for s, then f and g' approximately hide the hidden shift $s - t$. If we convert f and g to a function $h : D_N \to S$, then apply its dilation U_h with input $|D_N\rangle$ and discard the output, the result is a state $\rho_h = \rho_{f,g'}$ which is close to the state $\rho_{D_N/H}$ used in Algorithm 5.1.

We need to quantify how close. The relevant metric on states for us is the trace distance [18, section 9.2]. In general if ρ and ρ' are two states on a Hilbert space $\mathcal{H}$, the trace distance $||\rho - \rho'||$ is the maximum probability that any measurement, indeed any use in a quantum algorithm, will distinguish them. In our case,

$$||\rho_h - \rho_{D_N/H}|| = \frac{|s - t|}{N}.$$

If

$$\frac{|s - t|}{N} = 2^{-O(\sqrt{\log\ N})},$$

then with bounded probability, Algorithm 5.1 will never see the difference between ρ_h and $\rho_{D_N/H}$. Thus $2^{O(\sqrt{\log\ N})}$ guesses for s suffice. □

A second warm-up to the general case of Theorem 7.1 is the special case $A = \mathbb{Z}$. Recall that more computation is allowed for longer output. Suppose that the output

has n bits, i.e., the shift s is at most 2^n. In the language of deterministic hiding, we restrict the domain of $f, g : \mathbb{Z} \to S$ to the set $\{0, 1, 2, \ldots, 2^m\}$, where $m = n + \Theta(\sqrt{n})$, and interpret this set as $\mathbb{Z}/2^m$. Then f and g approximately differ by the shift s. If we form the state $\rho_{f,g}$ as in the proof of Corollary 7.2, then its trace distance from the state $\rho_{D_N/H}$, with $N = 2^m$, is $2^{-O(\sqrt{n})}$. Thus Algorithm 5.1 will never see the states differ.

Sketch of proof of Theorem 7.1. In the general case, the classification of finitely generated abelian groups says that

$$A \cong \mathbb{Z}^b \oplus \mathbb{Z}/N_1 \oplus \mathbb{Z}/N_2 \oplus \cdots \oplus \mathbb{Z}/N_a.$$

Assuming a bound on the length of the output, we can truncate each $\mathbb{Z}$ summand of A, as in the case $A = \mathbb{Z}$. (We suppose that we know how many bits of output are allocated to each free summand of A.) Thus we can assume that

$$A = \mathbb{Z}/N_1 \oplus \mathbb{Z}/N_2 \oplus \cdots \oplus \mathbb{Z}/N_a,$$

and the problem is to find s in time $2^{O(\sqrt{\log\ |A|})}$. In other words, the problem is to solve HSP for a finite group D_A.

The general element of D_A can be written $y^t x^a$ with $t \in \mathbb{Z}/2$ and $a \in A$. Following the usual first step, we can first prepare the state $\rho_{D_A/H}$. Then we can perform a quantum Fourier transform on each factor of A, then measure the answer, to obtain a label

$$k = (k_1, k_2, \ldots, k_a) \in A$$

and a qubit state

$$|\psi_k\rangle \propto |0\rangle + e^{2\pi i \sum_j s_j k_j / N_j} |1\rangle.$$

(As in section 4, this state is H-invariant in a two-dimensional representation V_k of D_A.) We will outline a sieve algorithm to compute any one coordinate of the slope, without loss of generality s_a.

As in Algorithm 5.3, we will guide the behavior of the sieve by an objective function α on A. Given k, let $b(k)$ be the first j such that $k_j \neq 0$. If $b < a$, then let

$$\alpha(k) = \sum_{j=1}^{b} \lceil 1 + \log_2(N_j + 1) \rceil - \lceil \log_2(k_b + 1) \rceil.$$

If $b = a$, then let

$$\alpha(k) = \sum_{j=1}^{a} \lceil 1 + \log_2(N_j + 1) \rceil.$$

As in Algorithm 5.3, we produce a list L of $2^{O(\sqrt{\log\ |A|})}$ qubits with states $|\psi_k\rangle$. Within the minimum of α on L, we repeatedly find pairs $|\psi_k\rangle$ and $|\psi_\ell\rangle$ that maximize $\alpha(k + \ell)$ or $\alpha(k - \ell)$, then we extract $|\psi_{k+\ell}\rangle$ from each such pair. The end result is a list of qubit states $|\psi_k\rangle$ with

$$k = (0, 0, \ldots, 0, k_a).$$

The set of k of this form is closed under sums and differences, so we can switch to Algorithm 5.1 to eventually determine the slope s_a. □

Note that many abelian groups A are not very different from cyclic groups, so that the generalized dihedral group D_A can be approximated for our purposes by a standard dihedral group. For example, if $A \cong \mathbb{Z}^a$ is free abelian with many bits of output allocated to each coordinate, then we can pass to a truncation

$$\mathbb{Z}/N_1 \oplus \mathbb{Z}/N_2 \oplus \cdots \oplus \mathbb{Z}/N_a$$

with relatively prime N_j's. In this case the truncation is cyclic.

8. Hidden subgroup generalities. In this section we will make some general observations about quantum algorithms for hidden subgroup problems. Our comments are related to work by Hallgren, Russell, and Ta-Shma [10] and by Grigni et al. [9].

8.1. Quantum oracles. The first step of all quantum algorithms for the hidden subgroup problem is to form the state $\rho_{G/H}$, or an approximation when G is infinite, except when the oracle $f : G \to S$ has special properties.

Suppose that a function $f : G \to S$ that hides the subgroup H. We can say that f *deterministically* hides H because it is a deterministic function. Some problems in quantum computation might reduce to a nondeterministic oracle $f : G \to \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space. We say that such an f *orthogonally* hides H if f is constant on each right coset Ha of H and orthogonal on distinct cosets. If a quantum algorithm invokes the dilation D_f of f and then discards the output, then it solves the orthogonal hidden subgroup problem as well as the deterministic one.

Computing D_f and discarding its output can also be viewed as a quantum oracle. A general quantum computation involving both unitary and nonunitary actions can be expressed as a quantum operation [18, Chapter 8]. In this case the operation is a map $\mathcal{E}_{G/H}$ on $\mathcal{M}(\mathbb{C}[G])$, where in general $\mathcal{M}(\mathcal{H})$ denotes the algebra of operators on a Hilbert space $\mathcal{H}$. It is defined by

$$\mathcal{E}_{G/H}(|a\rangle\langle b|) = \begin{cases} |a\rangle\langle b| & \text{if } Ha = Hb, \\ 0 & \text{if } Ha \neq Hb. \end{cases}$$

We say that the quantum oracle $\mathcal{E}_{G/H}$ *projectively* hides the subgroup H. Unlike deterministic and orthogonal oracles, the projective oracle is uniquely determined by H. Again, all quantum algorithms for hidden subgroup problems work with this more difficult oracle.

Finally, if G is finite, the projective oracle $\mathcal{E}_{G/H}$ can be applied to the constant pure state $|G\rangle$ to produce the state

$$\rho_{G/H} = \frac{|H|}{|G|} \sum |Ha\rangle\langle Ha|.$$

So an algorithm could use a no-input oracle that simply broadcasts copies of $\rho_{G/H}$. Such an oracle *coherently* hides H. This oracle has been also been called the random coset oracle [20] because the state $\rho_{G/H}$ is equivalent to the constant pure state $|Ha\rangle$ on a uniformly randomly chosen coset. Almost all existing quantum algorithms for finite hidden subgroup problems need only copies of the state $\rho_{G/H}$. Algorithms 3.1 and 5.3 are exceptions: They use $\rho_{D_N/H}$ to find the parity of the slope s and then rely on $\mathcal{E}_{D_N/H}$ with other inputs (constant pure states on subgroups) for later stages.

The possibly slower algorithm, Algorithm 5.1, works with the coherent oracle; it uses only $\rho_{D_N/H}$.

The distinctions between deterministic, orthogonal, and projective hiding apply to any hidden partition problem. In one special case, called the hidden stabilizer problem [14], a group G acts transitively on a set S and a function $f : S \to T$ is invariant under a subgroup $H \subseteq G$. The hidden stabilizer problem has enough symmetry to justify consideration of coherent hiding. It would be interesting to determine when one kind of hiding is harder than another. For example, if f is injective save for a single repeated value, then there is a sublinear algorithm for deterministic hiding [6]. But projective hiding requires at least linear time and we do not know an algorithm for coherent hiding which is faster than quadratic time.

In a variant of coherent HSP, the oracle outputs nonuniform mixtures of coset states $|Ha\rangle$. The mixtures may even be chosen adversarially. This can make the subgroup H less hidden, for example, in the trivial extreme in which the state is $|H\rangle$ with certainty. At the other extreme, we can always uniformize the state by translating by a random group element. Thus uniform coherent HSP is the hardest representative of this class of problems.

8.2. The character measurement. The second step of all quantum algorithms for the generic hidden subgroup problem is to perform the character measurement. (The measurement in our algorithms is only trivially different.) The result is the name or character of an irreducible unitary representation (or irrep) V and a state in V. Mathematically the character measurement is expressed by the Burnside decomposition of the group algebra $\mathbb{C}[G]$ as a direct sum of matrix algebras [21]:

$$\mathbb{C}[G] \cong \bigoplus_V \mathcal{M}(V).$$

Here $\mathcal{M}(V)$ is the algebra of operators on the irrep V; the direct sum runs over one representative of each isomorphism type of unitary irreps. The group algebra $\mathbb{C}[G]$ has two commuting actions of G, given by left and right multiplication, and with respect to these two actions,

$$\mathcal{M}(V) \cong V \otimes V^*,$$

so that the Burnside decomposition can also be written

$$\mathbb{C}[G] \cong \bigoplus_V V \otimes V^*. \tag{5}$$

In light of the identification with matrices, the factor of V^* is called the *row space*, while the factor of V is the *column space*.

The Burnside decomposition is also an orthogonal decomposition of Hilbert spaces and so corresponds to a projective measurement on $\mathbb{C}[G]$. This is the character measurement. A character transform is an orthonormal change of basis that refines equation (5). Its precise structure as a unitary operator depends on choosing a basis for each V.

The state $\rho_{G/H}$ has an interesting structure with respect to the Burnside decomposition. In general if $\mathcal{H}$ is a finite-dimensional Hilbert space, let $\rho_{\mathcal{H}}$ denote the uniform mixed state on $\mathcal{H}$; if V is a representation of a group G, let V^G denote its invariant space. It is easy to check that

$$\rho_{G/H} = \rho_{\mathbb{C}[G]^H},$$

where G (and therefore H) acts on $\mathbb{C}[G]$ by left multiplication. In the Burnside decomposition, the left multiplication action on each $V \otimes V^*$ is trivial on the right factor V^* and is just the defining action of G on V. Since $\rho_{G/H}$ is the uniform state on all H-invariant vectors in $\mathbb{C}[G]$, this property descends through the Burnside decomposition:

$$\rho_{G/H} = \bigoplus_V \rho_{V^H} \otimes \rho_{V^*}.$$

This relation has two consequences. First, as has been noted previously [9], the state on the row space V^* has no useful information. Second, since $\rho_{G/H}$ decomposes as a direct sum with respect to the Burnside decomposition, the character measurement sacrifices no coherence to the environment; it only measures something that the environment already knows. Our reasoning here establishes the following proposition.

PROPOSITION 8.1. *Let G be a finite group and assume an algorithm or oracle to compute the character transform on $\mathbb{C}[G]$. Then a process provides the state $\rho_{G/H}$ is* equivalent *to a process that provides the name of an irrep V and the state ρ_{V^H} with probability*

$$P[V] = \frac{(\dim V)(\dim V^H)|H|}{|G|}.$$

Proposition 8.1 sharpens the motivation to work with irreps in the hidden subgroup problem. If you obtain the state $\rho_{G/H}$, and if you can efficiently perform the character measurement on states, then you might as well apply it to $\rho_{G/H}$.

Proposition 8.1 and the definition of coherent HSP in section 8.1 suggest another class of oracles related to the hidden subgroup problem. In general an oracle might provide the name of a representation V and a state ρ which is some mixture of H-invariant pure states in V. It is tempting to describe such a ρ as H-invariant, but technically that is a weaker condition that also applies to other states. For example, the uniform state on V is H-invariant. So we say that ρ is *purely H-invariant* if it is supported on the H-invariant space V^H. For example, the uniform state $\rho_{G/H}$ is purely H-invariant. More generally the purely H-invariant states on $\mathbb{C}[G]$ are exactly the mixtures of constant pure states of right cosets $|Ha\rangle$.

PROPOSITION 8.2. *Let G be a finite group. Then any purely H-invariant state ρ on $\mathbb{C}[G]$ can be converted to $\rho_{G/H}$. In the presence of an algorithm or oracle to perform the character transform on $\mathbb{C}[G]$, any purely H-invariant state ρ on any irrep V can be converted to $\rho_{G/H}$.*

Proof. If we right-multiply ρ by a uniformly random element of G, it becomes $\rho_{G/H}$. If we perform the reverse character transform to a purely H-invariant state ρ on V, it becomes a purely H-invariant state on $\rho_{G/H}$ itself. □

The message of Proposition 8.2 is that the uniform mixture $\rho_{G/H}$ reveals the least information about H among all mixtures of coset states $|Ha\rangle$. The distribution on irreps V described in Proposition 8.1, together with the uniform state on V^H, also reveals the least information about H among all such distributions.

9. A general algorithm. In this section we will discuss a general algorithm for coherent HSP for an arbitrary finite group G and an arbitrary subgroup H. It is an interesting abstract presentation of all the algorithms for dihedral groups in this paper. Unfortunately, it might not be directly useful for any groups other than dihedral groups.

The algorithm uses the definitions and methods of section 8.2, together with a generalized notion of summand extraction. In general if V and W are two unitary representations of G, their tensor product decomposes as an orthogonal direct sum of irreps with respect to the diagonal action of G:

$$V \otimes W \cong \bigoplus_X \mathcal{H}_X^{W,V} \otimes X. \tag{6}$$

Here again the direct sum runs over one representative of each isomorphism class of irreps. The Hilbert space $\mathcal{H}_X^{W,V}$ is the *multiplicity factor* of the decomposition; its dimension is the number of times that X arises as a summand of $V \otimes W$. The decomposition defines a partial measurement of the joint Hilbert space $V \otimes W$, which extracts X (and $\mathcal{H}_X^{W,V}$). If V and W carry purely H-invariant states, then the state on X is also purely H-invariant.

ALGORITHM 9.1. *Input: An oracle that produces $\rho_{G/H}$.*

1. *Make a list L of copies of $\rho_{G/H}$. Extract an irrep V with a purely H-invariant state from each copy.*
2. *Choose an objective function α on* $\mathrm{Irrep}(G)$*, the set of irreps of G.*
3. *Find a pair of irreps V and W in L such that $\alpha(V)$ and $\alpha(W)$ are both low, but such that α is significantly higher for at least one summand of $V \otimes W$. Extract an irreducible summand X from $V \otimes W$ and replace V and W in L with X. Discard the multiplicity factor.*
4. *Repeat step* 3 *until α is maximized on some irrep V. Perform tomography on V to reveal useful information about H.*
5. *Repeat steps* 2–4 *to fully identify H.*

For any given group G, Algorithm 9.1 requires subalgorithms to compute the character measurement (5) and the tensor decomposition measurement (6). Efficient algorithms for character measurements and character transforms are a topic of active research [4, 16] and are unknown for many groups. We observe that tensor decomposition measurement at least reduces to the character measurement.

PROPOSITION 9.2. *Let V and W be irreducible representations of a finite group G. If group operations in G and summand extraction from $\mathbb{C}[G]$ are both efficient, then summand extraction from $V \otimes W$ is also efficient.*

Proof. Embed V and W into separate copies of $\mathbb{C}[G]$ in a G-equivariant way. Then apply the unitary operator

$$U(|a\rangle \otimes |b\rangle) = |b^{-1}a\rangle \otimes |b\rangle$$

to $\mathbb{C}[G] \otimes \mathbb{C}[G]$. The operator U transports left multiplication by the diagonal subgroup $G_\Delta \subset G \times G$ to left multiplication by G on the right factor. Then summand extraction from the right factor of $\mathbb{C}[G] \otimes \mathbb{C}[G]$ is equivalent to summand extraction from $V \otimes W$, since, after U is applied, the group action on the right factor of $\mathbb{C}[G] \otimes \mathbb{C}[G]$ coincides with the diagonal action on $V \otimes W$. □

In light of Beals's algorithm to compute a character transform on the symmetric group [4] and Proposition 9.2, Algorithm 9.1 may look promising when $G = S_n$ is the symmetric group. But the algorithm seems to work poorly for this group, because the typical irrep V of S_n is very large. Consequently the decomposition (6) typically involves many irreps of S_n. This offers very little control for a sieve.

Note that if Algorithm 9.1 were useful for the symmetric group, its time complexity would be $2^{O(\sqrt{\log |G|})}$ at best. This is the same complexity class as a known classical algorithm for the graph isomorphism or automorphism problem [3], which

is the original motivation for the symmetric hidden subgroup problem (SHSP). We believe that general SHSP is actually much harder than graph isomorphism. If graph isomorphism does admit a special quantum algorithm, it could be analogous to a quantum polynomial time algorithm found by van Dam, Hallgren, and Ip [24] for certain special abelian hidden shift problems. (In particular their algorithm applies to the Legendre symbol with a hidden shift.) All these problems have special oracles f that allow faster algorithms.

One reason that SHSP looks hard is that symmetric groups have many different kinds of large subgroups. For example, if $p_1, p_2, \ldots, p_n$ is a set of distinct primes, then

$$D_{p_1 p_2 \ldots p_n} \hookrightarrow S_{p_1+p_2+\cdots+p_n}$$

(exercise). Thus DHSP reduces to SHSP. Hidden shift in the symmetric group also reduces to SHSP (exercise).

The sieve of Algorithm 9.1 looks the most promising when the group G is large but $V \otimes W$ always has few terms. This is similar to demanding that most or all irreps of G are low-dimensional. So suppose that all irreps have dimension at most k and consider the limit $|G| \to \infty$ for fixed k. Isaacs and Passman [12] showed that there is a function $f(k)$ such that if all irreps have dimension at most k, then G has an abelian subgroup $\exp(A)$ of index at most $f(k)$. By the reasoning of Proposition 2.1, the hardest hidden subgroup H for a such a G is one which is disjoint from $\exp(A)$ (except for the identity). But by the reasoning of section 6, any such hidden subgroup problem reduces to the hidden shift problem on A. The generalized sieve of Algorithm 9.1 is not as fast as the dihedral sieve on D_A.

Acknowledgments. Some elements of the algorithms in this article are due to Ettinger and Høyer [7]. Regev has presented some related ideas related to lattice problems [20] and more recently has found a space-efficient variation of the algorithms in this article [19]. (We have also borrowed some aspects of his exposition of our algorithm.) The author would like to thank Robert Beals, Robert Guralnick, Peter Høyer, and Eric Rains for useful discussions. The also would also like to thank the referees for useful comments.

REFERENCES

[1] M. Ajtai, R. Kumar, and D. Sivakumar, *A sieve algorithm for the shortest lattice vector problem*, in Proceedings of the 33rd Annual ACM Symposium on Theory of Computing, 2001, pp. 601–610.

[2] M. Artin, *Algebra*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1991.

[3] L. Babai and E. M. Luks, *Canonical labeling of graphs*, in Proceedings of the 15th Annual ACM Symposium on Theory of Computing, ACM Press, New York, 1983, pp. 171–183.

[4] R. Beals, *Quantum computation of Fourier transforms over symmetric groups*, in ACM Symposium on Theory of Computing, 1997, pp. 48–53.

[5] A. Blum, A. Kalai, and H. Wasserman, *Noise-tolerant learning, the parity problem, and the statistical query model*, J. ACM, 50 (2003), pp. 506–519.

[6] H. Buhrman, C. Dürr, M. Heiligman, P. Høyer, F. Magniez, M. Santha, and R. de Wolf, *Quantum algorithms for element distinctness*, in IEEE Conference on Computational Complexity, 2001, pp. 131–137.

[7] M. Ettinger and P. Høyer, *On quantum algorithms for noncommutative hidden subgroups*, Adv. in Appl. Math., 25 (2000), pp. 239–251.

[8] M. Ettinger, P. Høyer, and E. Knill, *Hidden subgroup states are almost orthogonal*.

[9] M. Grigni, L. J. Schulman, M. Vazirani, and U. V. Vazirani, *Quantum mechanical algorithms for the nonabelian hidden subgroup problem*, in ACM Symposium on Theory of Computing, 2001, pp. 68–74.

[10] S. Hallgren, A. Russell, and A. Ta-Shma, *Normal subgroup reconstruction and quantum computation using group representations*, in ACM Symposium on Theory of Computing, 2000, pp. 627–635.
[11] P. Høyer, *personal communication*, 2003.
[12] I. M. Isaacs and D. S. Passman, *Groups with representations of bounded degree*, Canad. J. Math., 16 (1964), pp. 299–309.
[13] G. Ivanyos, F. Magniez, and M. Santha, *Efficient quantum algorithms for some instances of the non-abelian hidden subgroup problem*, Internat. J. Found. Comput. Sci., 14 (2003), pp. 723–739.
[14] A. Kitaev, *Quantum measurements and the abelian stabilizer problem*, arXiv:quant-ph/9511026.
[15] G. Kuperberg. `dhspsim.py`, included with the source of arXiv:quant-ph/0302112.
[16] C. Moore, D. Rockmore, and A. Russell, *Generic quantum fourier transforms*, in Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, 2004, pp. 778–787.
[17] M. Mosca and C. Zalka, *Exact quantum fourier transforms and discrete logarithm algorithms*, Int. J. Quantum Inf., 2 (2004), pp. 91–100.
[18] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, UK, 2000.
[19] O. Regev, *A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space*, arXiv:quant-ph/0406151.
[20] O. Regev, *Quantum computation and lattice problems*, SIAM J. Comput., 33 (2004), pp. 738–760.
[21] J.-P. Serre, *Linear Representations of Finite Groups*, Graduate Texts in Math. 42, Spring-Verlag, New York, 1977.
[22] P. W. Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM J. Comput., 26 (1997), pp. 1484–1509.
[23] D. R. Simon, *On the power of quantum computation*, SIAM J. Comput., 26 (1997), pp. 1474–1483.
[24] W. van Dam, S. Hallgren, and L. Ip, *Quantum algorithms for some hidden shift problems*, in Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms, Baltimore, MD, 2003, SIAM, Philadelphia, 2003, pp. 489–498.