The Abelian Hidden Subgroup Problem

Stephen McAdam
Department of Mathematics
University of Texas at Austin
mcadam@math.utexas.edu

Introduction: The general Hidden Subgroup Problem is as follows. Let G be a known finite group. Let H be an unknown subgroup of G. Suppose there is a set S and a function $f : G \to S$ such that for $g, k \in G$, $f(g) = f(k)$ if and only if g and k are in the same coset of H. (We say f separates cosets.) The question is to find a method of using f to determining what H is, significantly faster than naïve methods. The most naïve method is to compare every f(g) to f(identity). Then g will be in H exactly when equality holds. Of course, standard facts about group theory, such as LaGrange's theorem, can clearly be used to speed that up somewhat, but not really significantly.

These notes outline how quantum computations can be used to attack the problem in the case that G is Abelian. Later, we will discuss Kiteav's reduction of the discrete log problem to the Abelian hidden subgroup problem. We will also discuss the oft made statement that the factorization problem is a special case of the Abelian hidden subgroup problem, seeing that it is true if we modify our definition slightly.

Of the few quantum algorithm's I have learned, the one discussed here is the easiest, and so would be a good place for a beginner to start. However, for a true beginner, I would strongly recommend first reading section 2 of [M1], which gives a general introduction to the physics and notation relevant to quantum computing. If one follows that by then reading this, and then reading the rest of [M1], and finally reading [M3] (the most elaborate quantum algorithm I know), the reader will have learned everything I know about the subject (and will no doubt realize how little that is).

These notes are not complete. They present the parts that I find prettiest, namely, a review of group characters, the group Fourier transform, and the actual quantum computation. They do not discuss the efficiency of the method. These notes are adapted from [L], which discusses the details missing here, and also discusses what progress has been made on the non-Abelian case.

Our computations will on the complex inner-product space having the set of symbols $\{|g, s\rangle \mid g \in G, s \in S\}$ as an orthonormal bases. The inner product will be the standard one. Recall, a unit vector in such an inner product space is called a state, and (mathematical) states accurately represent actual (physical) states of a collection of qubits. (We will assume the reader is comfortable with the meaning of expressions such as $\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g, f(g)\rangle$, and their relation to physical reality.)

BRIEF OUTLINE (details later): Let G be Abelian. Group character theory defines a subgroup $H^{\perp}$ that is determined by H. If H is known, $H^{\perp}$ can be found. Of course, we do not know H. However, the quantum computations we will present allow us to find $H^{\perp}$ directly. Knowing $H^{\perp}$, we can then find $H^{\perp\perp}$. We will prove that $H^{\perp\perp} = H$, and so be done. Well, actually, it is a bit more complicated. Each run of the computer will be guarantied to produce an element of $H^{\perp}$, all possibilities being equally likely to appear. We must collect a set of such elements, and hope that they constitute a generating set for $H^{\perp}$. If so, using them, we can find H. (See [L] for discussions of the efficiency of finding a generating set for $H^{\perp}$, and a method of using such to find H.)

Of course, we need matters to be in a computational form. For instance, we will assume $G = Z_{N_1} \oplus Z_{N_2} \oplus \cdots \oplus Z_{N_w}$ (recalling that any finite Abelian group is isomorphic to such). That allows us to use a register in the computer to input an element g of G, using the standard "base 2" method. (Example: If (3 mod 5, 1 mod 4) ε $Z_5 \oplus Z_4$, that element could be expressed as (0, 1, 1, 0 1), the first 011 representing 3, and the last 01 representing 1, where the first part has three places, since $2^3 \geq 5 > 2^2$. What this does is encode $Z_5 \oplus Z_4$ as a subset of Z_2^{3+2}.) We will also assume a second register of the computer can be used to deal with f(g), which means we need to be able to (efficiently) identify S with a subset of Z_2^s, for some s. (If, say s = 3, and $G = Z_5 \oplus Z_4$, we would need 3 + 2 + 3 qubits in our computer.) Finally, we must assume the function f can be efficiently computed.

Our computer will have two registers, the first dealing with the g ε G, the second dealing with the s ε S. Our computer program will involve three steps. Near the end of section 2 of [M1], it is shown that there is a quantum gate that converts $\frac{1}{\sqrt{|G|}} \sum_{g \varepsilon G} |g,0>$ to $\frac{1}{\sqrt{|G|}} \sum_{g \varepsilon G} |g, f(g)>$. Therefore, that conversion is an allowable quantum calculation, and is the middle one of our three steps. It is the only one of the three that deals with both registers simultaneously.

The first and last steps in our program only deal with the first register. Actually, both those steps are the same, namely, applying the group Fourier transform. In order to introduce that transform, we must first remind the reader of the theory of (irreducible) characters of a finite Abelian group. (The knowledgeable reader can skip ahead.)

Definition: An irreducible character of the Abelian group G is a homomorphism from G to the multiplicative group of nonzero complex numbers. (Whenever we use the word character, it will be understood to refer to an irreducible character of G.) $\overline{G}$ will denote the set of all such characters.

$\overline{G}$ is easily made into a group. The identity is the character that sends each element of G to 1. The inverse of χ sends g to the inverse of $\chi(g)$. The operation is function multiplication.

Since G is a group under addition, if χ is a character and g, h ε G, then $\chi(g + h) = \chi(g)\chi(h)$. Thus, for n an integer, we have $\chi(ng) = (\chi(g))^n$. If it happens that ng = 0, then $1 = \chi(0) = \chi(ng) = (\chi(g))^n$, and so $\chi(g)$ is an n-th root of unity. (In particular, $\chi(g)$ is an n-th root of unity for n the order of g.)

Notation: We have $G = Z_{N_1} \oplus Z_{N_2} \oplus \cdots \oplus Z_{N_w}$. For $a = (a_1, a_2, \ldots, a_w)$ and $b = (b_1, b_2, \ldots b_w)$ in G (and treating the a_j and b_j as integers between 0 and $N_j - 1$), define $\chi_a(b)$ to be $\prod_{j=1}^{w} \exp(2\pi i a_j b_j / N_j)$.

(A) Lemma: The map $a \rightarrow \chi_a$ is a group isomorphism between G and $\overline{G}$.

Proof: It is straightforward to see that χ_a is a character. Now let χ be any character. For $1 \leq j \leq w$, let B_j ε G be the w-tuple consisting entirely of zeros except for the j-th position in which there is a 1. Now $N_j B_j = 0$, and so $\chi(B_j)$, being an N_j-th root of unity, has the form $\exp(2\pi i a_j / N_j)$ for some integer a_j between 0 and $N_j - 1$. Since $(b_1, b_2, \ldots, b_w) = b_1 B_1 + b_2 B_2 + \ldots + b_w B_W$, we easily see that $\chi = \chi_a$, for $a = (a_1, a_2, \ldots, a_w)$. Thus $\overline{G} = \{\chi_a \mid a \in G\}$, and so our map is onto. Furthermore, it is easily verified that it is a group homomorphism. Now if χ_a is the identity of $\overline{G}$, then $\chi_a(B_j) = 1$. As it also equals $\exp(2\pi i a_j / N_j)$, we must have $N_j | a_j$, so that a_j is the zero of Z_{Nj}. As that holds for $1 \leq j \leq w$, we must have a = 0 in G. We therefore see that the kernel of our map is {0}, and we are done.

(B) Lemma: $\chi_a(b) = \chi_b(a)$. Also, $\chi_{-a}(b) = \overline{\chi_a(b)}$ (with the overbar denoting complex conjugation).

Proof: The first statement is immediate from the definition. As for the second,

$$\chi_{-a}(b) = \prod_{j=1}^{w} \exp(-2\pi i a_j b_j / N_j) = \overline{\prod_{j=1}^{w} \exp(2\pi i a_j b_j / N_j)} = \overline{\chi_a(b)}.$$

(C) Lemma: Let a,c ε G. Then $\sum_{b \varepsilon G} \chi_a(b)\overline{\chi_c(b)}$ equals 0 unless a = c, in which case it equals |G|.

Proof: $\sum_{b \varepsilon G} \chi_a(b)\overline{\chi_c(b)} = \sum_{b \varepsilon G} \left(\left(\prod_{j=1}^{w} \exp(2\pi i a_j b_j / N_j) \right) \left(\overline{\prod_{j=1}^{w} \exp(2\pi i c_j b_j / N_j)} \right) \right) =$

$\sum_{b \varepsilon G} \left(\left(\prod_{j=1}^{w} \exp(2\pi i a_j b_j / N_j) \right) \left(\prod_{j=1}^{w} \exp(-2\pi i c_j b_j / N_j) \right) \right)$. Letting $d_j = a_j - c_j$, that becomes

$\sum_{b \varepsilon G} \left(\left(\prod_{j=1}^{w} \exp(2\pi i d_j / N_j) \right)^{b_j} \right) = \sum_{b_1 \varepsilon Z_{N_1}} \sum_{b_2 \varepsilon Z_{N_2}} \cdots \sum_{b_w \varepsilon Z_{N_w}} \left(\left(\prod_{j=1}^{w} \exp(2\pi i d_j / N_j) \right)^{b_j} \right) =$

$\prod_{j=1}^{w} \left(\sum_{b_j \varepsilon Z_{N_j}} (\exp(2\pi i d_j / N_j)^{b_j} \right)$. If N_j divides d_j, then $\sum_{b_j \varepsilon Z_{N_j}} (\exp(2\pi i d_j / N_j)^{b_j} = N_j$. Otherwise, it is a geometric series summing to $\frac{1 - (\exp(2\pi i d_j / N_j))^{N_j}}{1 - \exp(2\pi i d_j / N_j)}$, which equals 0, since the numerator is 0 but the denominator is not. Therefore, $\sum_{b \varepsilon G} \chi_a(b) \overline{\chi_c(b)} = 0$ unless N_j divides d_j for $1 \leq j \leq w$, in which case it equals $\prod_{j=1}^{w} N_j = |G|$. However, since $d_j = a_j - c_j$ is in Z_{N_j}, we see that N_j divides d_j for $1 \leq j \leq w$, exactly when $a = c$.

Remark: (C) is called the first orthogonality relation, and holds for the irreducible characters of non-Abelian groups as well. Since the definition of an irreducible character is harder for the non-Abelian case, the proof is somewhat harder. See [I, Chapter 2].

(D) Corollary: $\sum_{b \varepsilon G} \chi_a(b)$ equals 0 unless $a = 0$, in which case it equals $|G|$.

Proof: This is just the special case of (C) in which $c = 0$ (since χ_0 is identically 1).

We previously mentioned that the first and last steps of our program will be an application of the group Fourier transform to the first register (that register being used to deal with states which are linear combinations of basic states $|k\rangle$ for $k \varepsilon G$).

Notation: Let H(G) be the complex inner-product space having the set of symbols $\{|k\rangle \mid k \varepsilon G\}$ as an orthonormal bases.

Definition: The Fourier transform of G is the operator on H(G) which takes the basis element $|k\rangle$ to $\frac{1}{\sqrt{|G|}} \sum_{g \varepsilon G} \chi_g(k) \mid g >$.

(E) Lemma: F is a unitary operator (and so applying it constitutes an allowable quantum computation).

Proof: We must show that {F(|k>) | k ε G} is an orthonormal bases of H(G). For k, h ε G, the inner product of f(|h>) and f(|k>) is $\frac{1}{|G|}\sum_{g\varepsilon G}\chi_g(k)\overline{\chi_g(h)}$ = (by (B)) $\frac{1}{|G|}\sum_{g\varepsilon G}\chi_k(g)\overline{\chi_h(g)}$.
By (C), that is 0 unless h = k, in which case it is 1. The lemma follows (linear independence using a standard argument).

The next lemma is not strictly required, but is of interest.

(F) Lemma: For k ε G, F(F(|k>)) = |-k>.

Proof: F(F(|k>)) = $\frac{1}{\sqrt{|G|}}\sum_{g\varepsilon G}\chi_g(k)F(|g>)$ =

$\frac{1}{\sqrt{|G|}}\sum_{g\varepsilon G}\chi_g(k)[\frac{1}{\sqrt{|G|}}\sum_{-h\varepsilon G}\chi_{-h}(g)\,|-h>]$ =

$\frac{1}{|G|}\sum_{-h\varepsilon G}[\sum_{g\varepsilon G}\chi_g(k)\chi_{-h}(g))]\,|-h>$ = (using (B))

$\frac{1}{|G|}\sum_{-h\varepsilon G}[\sum_{g\varepsilon G}\chi_k(g)\overline{\chi_h(g)}]\,|-h>$ = |-k> (using (C)).

Our brief outline said that we will find H by first finding $H^{\perp}$. We review that subgroup.

Definition: Let H be a subgroup of G. Then $H^{\perp}$ = {g ε G | χ_g(h) = 1 for all h ε H} = ∩Ker χ_h over all h ε H. (The last equality, by (B), makes it clear that $H^{\perp}$ is a subgroup of G).

(G) Lemma: $|H^{\perp}| = |G/H|$.

Proof: We will show more than what is stated, but we only need the statement.
If g ε $H^{\perp}$, then clearly H ⊆ Ker χ_g. Therefore, the map γ_g defined on G/H by $\gamma_g(a + H) = \chi_g(a)$ is well defined. Now γ_g is easily seen to be a homomorphism into the multiplicative group of nonzero complex numbers. That is, it is a character of G/H. Therefore, the map Γ sending g ε $H^{\perp}$ to γ_g is a map from $H^{\perp}$ to $\overline{G/H}$.

We claim Γ is a homomorphism. For g, k ε $H^{\perp}$, $\Gamma(g + k) = \gamma_{g+k}$ sends a + H to $\chi_{g+k}(a) = \chi_g(a)\chi_k(a)$, and so does $\Gamma(g)\Gamma(k)$.

We claim that Γ is one-to-one. Suppose g ε $H^{\perp}$ is in the kernel of Γ. Then for all a + H, we have $\chi_g(a) = \gamma_g(a + H) = 1$. This is true for all a ε G, and so $\chi_g = \chi_0$. By (A), g = 0.

We claim that Γ is onto. Pick some γ ε $\overline{G/H}$. Define χ on G by saying $\chi(a) = \gamma(a + H)$. Clearly χ is a character of G, and so $\chi = \chi_g$ for some g ε G. However, clearly H $\subseteq$ Ker χ, and so g ε $H^{\perp}$. Thus $\gamma = \gamma_g = \Gamma(g)$.

We now see that Γ is an isomorphism. In particular, $|H^{\perp}| = |\overline{G/H}|$. However, by (A), we already know $|\overline{G/H}| = |G/H|$, and so we are done.

Notation: For H a subgroup of G, let $|H> = \frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} |h>$ (which is a state).

(H) Lemma: If H is a subgroup of G, then $F(|H>) = |H^{\perp}>$.

Proof: $F(|H>) = \frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} F(|h>) = \frac{1}{\sqrt{|H||G|}} \sum_{h \varepsilon H} [\sum_{t \varepsilon G} \chi_t(h) |t>] =$

$\frac{1}{\sqrt{|H||G|}} \sum_{t \varepsilon G} (\sum_{h \varepsilon H} \chi_t(h))|t>$. Now χ_t is a character of G. However, when restricted to H, it is obviously also a character of H. Therefore, by (D), $\sum_{h \varepsilon H} \chi_t(h)$ is 0 unless χ_t restricted to H is identically 1, in which case that sum equals |H|. However, χ_t restricted to H is identically 1 exactly when t is in $H^{\perp}$. Therefore, we have

$F(|H>) = \frac{1}{\sqrt{|H||G|}} \sum_{t \varepsilon H^{\perp}} |H||t> =$ (using G) $\frac{1}{\sqrt{|H^{\perp}|}} \sum_{t \varepsilon H^{\perp}} |t> = |H^{\perp}>$.

We give two proofs of the next corollary.

(I) Corollary: If H is a subgroup of G, then $H^{\perp\perp} = H$.

Proof #1: The definition of $H^{\perp}$ shows $H \subseteq H^{\perp\perp}$. By two applications of (G), we have $|H^{\perp\perp}| = |G|/|H^{\perp}| = |G|/(|G|/|H|) = |H|$.

Proof #2: By (H), $F(F(|H>)) = |H^{\perp\perp}> = \frac{1}{\sqrt{|H^{\perp\perp}|}} \sum_{k \varepsilon H^{\perp\perp}} |k>$. However, by (F), we also have

$F(F(H)) = \frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} F(F(|h>)) = \frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} |-h> = \frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} |h>$. Since all the |k> and |h> are part of the same bases of our vector space H(G), the equality of these two linear combinations implies they are identical, and we are done.

Notation: For $t \varepsilon G$, let τ_t be the operator which sends the basis element |g> to |t+g>. (Clearly that operator permutes the canonical orthonormal basis elements of H(G), and so is unitary.) Also, let ϕ_t be the operator which multiplies the basis element |g> of H(G) by $\chi_g(t)$. (As the norm of $\chi_g(t)$ is 1, this operator is also unitary.) Finally, let T be a transversal for the subgroup H of G (i.e., T is a set of coset representatives).

(J) Lemma: For $t \varepsilon G$, we have $F\tau_t = \phi_t F$.

Proof: One can easily verify that both sides send the basis vector |g> of H(G) to $\frac{1}{\sqrt{|G|}} \sum_{k \varepsilon G} \chi_k(t+g) |k>$, (using that $\chi_k(t)\, \chi_k(g) = \chi_k(t+g)$).

We are now ready to beginning showing how to program the computer to produce equally likely random elements of $H^{\perp}$.

(K) The Program: We begin with the computer in the state |0,0>. We then apply F to the first register, which is easily seen to result in the state $\frac{1}{\sqrt{|G|}} \sum_{g \varepsilon G} |g,0>$. As explained earlier, there is a quantum gate that converts the above state to the state $\frac{1}{\sqrt{|G|}} \sum_{g \varepsilon G} |g, f(g)>$. Since f separates cosets, we know that for $t \varepsilon T$ and $h \varepsilon H$, f(t + h) = f(t). Therefore, (using that |G| = |T||H|) our previous state can be written as

$$\frac{1}{\sqrt{|T|}} \sum_{t \varepsilon T} \left(\frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} |t+h>\right) |f(t)> = \frac{1}{\sqrt{|T|}} \sum_{t \varepsilon T} \left(\frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} \tau_t(|h>)\right) |f(t)> =$$

$$\frac{1}{\sqrt{|T|}} \sum_{t \varepsilon T} \left(\tau_t\left(\frac{1}{\sqrt{|H|}} \sum_{h \varepsilon H} |h>\right)\right)|f(t)> = \frac{1}{\sqrt{|T|}} \sum_{t \varepsilon T} \left(\tau_t(|H>)\right)|f(t)>.$$ We next apply F to

the first register. Using (J), we see the result is $\frac{1}{\sqrt{|T|}}\sum_{t\varepsilon T}(\phi_t F(|H>))|f(t)>$, which by (H) equals $\frac{1}{\sqrt{|T|}}\sum_{t\varepsilon T}(\phi_t(|H^{\perp}>))|f(t)> = \frac{1}{\sqrt{|T|}}\sum_{t\varepsilon T}(\phi_t(\frac{1}{\sqrt{|H^{\perp}|}}\sum_{k\varepsilon H^{\perp}}|k>))|f(t)>$. Using the definition of ϕ_t and the fact that (G) shows $|H^{\perp}| = |T|$, this last state equals $\frac{1}{|T|}\sum_{t\varepsilon T}(\sum_{k\varepsilon H^{\perp}}\chi_k(t)|k>)|f(t)> = \frac{1}{|T|}\sum_{t\varepsilon T,k\varepsilon H^{\perp}}\chi_k(t)|k,f(t)>$. At this point, let us mention that since f separates cosets, f takes distinct values on distinct members of T. Thus, the various |k, f(t)> are $|H^{\perp}||T| = |T|^2$ distinct states. Since the norm of $\chi_k(t)$ is 1, if $\frac{1}{|T|}\sum_{t\varepsilon T,k\varepsilon H^{\perp}}\chi_k(t)|k,f(t)>$ is read, all of the states |k, f(t)> (with $k \varepsilon H^{\perp}$, $t \varepsilon T$) have an equally likely chance (namely $1/|T|^2$) of appearing. Also, any such k appears in exactly |T| such states, and so each $k \varepsilon H^{\perp}$ has a chance of exactly 1/|T| of being read from the first register. Therefore, we produce a random element of $H^{\perp}$, all equally likely.

Remark: There is a subtlety in the above mathematics that might be overlooked. Let us illuminate it. At the second step, we went from the state $\frac{1}{\sqrt{|G|}}\sum_{g\varepsilon G}|g,0>$ to the state $\frac{1}{\sqrt{|G|}}\sum_{g\varepsilon G}|g,f(g)>$. Superficially, that appears to leave the first register unaffected. However, the step actually has a profound affect on the first register, since it entangles it with the second register. If one reads the second register of $\frac{1}{\sqrt{|G|}}\sum_{g\varepsilon G}|g,0>$, one will assuredly get 0, and the first register will remain in the state $\frac{1}{\sqrt{|G|}}\sum_{g\varepsilon G}|g>$. However, if one reads the second register of $\frac{1}{\sqrt{|G|}}\sum_{g\varepsilon G}|g,f(g)>$, one will get some choice of f(g). Suppose g is in the coset t + H (with $t \varepsilon T$). Then by entanglement, the first register will be in the state $\frac{1}{\sqrt{|H|}}\sum_{g\varepsilon t+H}|g>$. Our process had three steps, the first and last being the application of F to the first register. We are discussing the middle step. Suppose it was left out. Then we would merely start with |0, 0>, and apply F twice to the first register. By (F), we would end up back at |0, 0>. It is the middle step, the entangling step, which works the magic. (Entanglement is thoroughly examined in [M4], and perhaps too thoroughly in [M2].)

The rest of the procedure consists of repeating (K) many times, each time getting a random element of $H^{\perp}$. Now [L, Theorem D1, appendix D] states that if $n + \lceil \log |K| \rceil$ elements are chosen at random from a finite group K, then the probability they generate K exceeds $1 - 1/2^n$. We wish to apply that to $K = H^{\perp}$. However, not knowing $|H^{\perp}|$, we instead use $n + \lceil \log |G| \rceil$. For adequately large n, we will almost certainly have a generating set for $H^{\perp}$. Next, [L, p. 23] shows how to use that generating set to successively find random elements of H. Again, we take $n + \lceil \log |G| \rceil$ of them, since we do not know |H|.

DISCRETE LOG PROBLEM: Let p be prime, and let b be a primitive root for p. Given x, it is easy to find c such that $c \equiv b^x \bmod p$. However, given c it is very hard to find x with $b^x \equiv c \bmod p$. That is called the discrete log problem. The $x \rightarrow b^x \bmod p$ function is an example of a one-way function, and such functions are prime candidates for use in cryptography. Indeed, El Gamel encryption makes use of the difficulty of finding x given c. In [S], along with his famous quantum factorization algorithm, Shor gives a lesser mentioned but equally interesting quantum algorithm for solving the discrete log problem. (It is also exhibited in [M1].)

Here, we give a solution to the discrete log problem due to Kiteav [K]. It is accomplished by converting the discrete log problem into an Abelian hidden subgroup problem.

(L) Let b be a primitive root of the prime p and suppose p does not divide c. We wish to find x with $b^x \equiv c \bmod p$. Let $G = Z_{p-1} \oplus Z_{p-1}$. Let f be the function from G to $S = U_p$, the multiplicative group of units mod p, defined by $f(u, v) = b^u c^v \bmod p$. Now f is easily seen to be a group homomorphism from G to U_p, and its kernel is $H = \{ (u, v) \mid f(u, v) = 1 \bmod p\}$. One easily sees that f separates cosets of H, the hidden subgroup we seek. Suppose an element (u, v) of H has been found. Then since $b^x \equiv c \bmod p$, we have $1 \equiv b^u c^v \equiv b^{u+vx} \bmod p$. As b is a primitive root for p, we have $u + vx \equiv 0 \bmod p - 1$. For each choice of v in $\{0, 1, 2, \ldots, p - 2\}$, there is one choice of u making that congruence true. Many of the v in that set are relatively prime to $p - 1$. Assuming that is so for our v, we see $x \equiv -uv^{-1} \bmod p - 1$, and we are done. (Note that we did not need to find a generating set for H. Finding v^{-1} from v is easy via the Euclidean algorithm.)

Remark: Applying the Abelian hidden subgroup algorithm to $Z_{p-1} \oplus Z_{p-1}$ would require using the Fourier transform of that group. I do not know how efficient that is, nor am I interested enough to try to learn. A more interested reader can find some discussion of such matters in [L].

FACTORING: One often sees claims that factorization can be reduced to the Abelian hidden subgroup problem. That appears to be correct under a weaker version of the hidden subgroup problem, as we now discuss.

Suppose we wish to factor n (not a prime power). An incorrect approach would be to say that we can write $n = kh$ with $GCD(k, h) = 1$. Thus $Z_n \approx Z_k \oplus Z_h$, and we wish to find the hidden subgroup $Z_k \oplus 0$ of Z_n. The problem here is that we do not have any coset separating f. Without the additional information given by f, our task is exactly as hard as finding a factor of n.

We must find an approach that incorporates some useful f.

Pick a random b with $1 < b < n$, and use the Euclidean algorithm to quickly find GCD(b, n). If the GCD exceeds 1, it is a non-trivial factor of n. The more likely case is that GCD equals 1. If so, let r (unknown) be the order of b mod n in the group of units of Z_n. Since r divides $\phi(n)$, let $H = \{0, r, 2r, \ldots, (\phi(n)/r - 1)r\} \subseteq Z_{\phi(n)} = G$. H is our hidden subgroup. Is there an f? Yes! For $x \in Z_{\phi(n)}$, let $f(x) = b^x \bmod n$. We then have f(x) = f(y) IFF $b^x \equiv b^y \bmod n$ IFF $r \mid x - y$ IFF x and y are in the same coset of H.

Suppose we could find H. Actually, we do not have to find all of H; a few random elements of H will probably suffice to factor n, as we now explain.

First of all, we hope that r is even. If the process we are about to explain fails to factor n within a fairly small number of tries, we can guess that for the b we selected, r is probably odd. In that case, we will pick a new b, and try again. (Exercise: Show that at least half of all b have an even r. Consider n as a product of prime powers, and use that prime powers have primitive roots.) Assuming r is even, suppose we have managed to find some h in H. Then $h = rk$ with $0 \le k \le \phi(n)/r - 1$.

Claim: There is a good chance that $GCD(b^{h/2} - 1, n)$ is a non-trivial factor of n. If k is even, then $r \mid h/2$, and we easily see that GCD equals n, and we fail. However, if k is odd, then r does not divide h/2, and so we know that $b^{h/2}$ is not congruent to 1 mod n. Thus $GCD(b^{h/2} - 1, n) < n$. Therefore, we must show there is a good chance that GCD is bigger than 1. Now $b^{h/2} - 1 = (b^{r/2})^k - 1$ has $b^{r/2} - 1$ as a factor. Thus, it will suffice to show there is a good chance that $GCD(b^{r/2} - 1, n) > 1$. In fact, the well-known exponential factorization method (see [M1, section 1] or many textbooks) shows the chance of that last GCD being greater than 1 is at least ½. Within a few tries for b and h, we should be able to factor n.

However, we cannot use the Abelian hidden subgroup algorithm described above to find elements of H. That algorithm assumes that G is known. Our $G = Z_{\phi(n)}$ is not known. We do not know $\phi(n)$. What we have is a variant of the hidden subgroup problem (as defined at the start of this paper). We have an unknown G, and unknown H, and a known f.

EXAMPLE: Suppose we do not know the factorization of 21, but are told by an oracle that $\phi(21) = 12$. Since GCD(2, 21) = 1, we try letting b = 2. We know that the order of 2 mod 21 (= r, unknown) will divide 12, and so the multiples of r (taken mod 12) will constitute a hidden subgroup H of Z_{12}. We also know the function $f(x) = 2^x \bmod 21$ is a coset separating function for H. (Note that f(5) = 11, since $2^5 \equiv 11 \bmod 21$.) Thus, the algorithm given above can be used to find H. Notice that at the end of the second step of the algorithm (the entangling step), the computer will be in the state

$$\frac{1}{\sqrt{12}}(|0, 1\rangle + |1, 2\rangle + |2, 4\rangle + |3, 8\rangle + |4, 16\rangle + |5, 11\rangle$$

$$+ |6, 1\rangle + |7, 2\rangle + |8, 4\rangle + |9, 8\rangle + |10, 16\rangle + |11, 11\rangle).$$

(Notice the two cycles of second entries, each of length 6.)

With a small amount of luck, the algorithm would soon tell us that $H = \{0, 6\}$, from which we would learn that the order of r is 6. Now using the exponential factorization

technique, we cross our fingers and find GCD($2^{6/2} - 1$, 21). It is 7, and so we have found a factor of 21.

REMARK: In the special case where n = pq is the product of two distinct primes, since n - φ(n) + 1 = p + q, if we know φ(n), we would have pq and p + q, and so could easily factor n.

But what if the oracle is closed for remodeling, and we do not know what ϕ(21) equals? Shor's famous algorithm ([S], [M1]) deals with that problem. Since $2^9 = 512$ is the first power of 2 larger than 21^2, it works with Z_{512}. (That modulus is manageably small, yet big enough to give a high probability of success to Shor's method.) We also use the function $f(x) = 2^x \bmod 21$, but now for all x in Z_{512}. However, f is not a coset separating function. We saw above that f gives cycles of length 6, and so were it a coset separating function, there would have to be 6 cosets, so that the subgroup they came from would have to have size 512/6 = 85 + 2/6. But that is not an integer. Instead, one step of Shor's algorithm has the computer in the state

$$(1/\sqrt{512})\big[|0, 1\rangle + |1, 2\rangle + |2, 4\rangle + |3, 8\rangle + |4, 16\rangle + |5, 11\rangle +$$

$$|6, 1\rangle + |7, 2\rangle + |8, 4\rangle + |9, 8\rangle + |10, 16\rangle + |11, 11\rangle +$$

$$|12, 1\rangle + |13, 2\rangle + |14, 4\rangle + |15, 8\rangle + |16, 16\rangle + |17, 11\rangle +$$

$$\vdots$$

$$|504, 1\rangle + |505, 2\rangle + |506, 4\rangle + |507, 8\rangle + |508, 16\rangle + |509, 11\rangle +$$

$$|510, 1\rangle + |511, 2\rangle\big].$$ (There are 85 complete cycles, and 2/6 of another cycle.)

Suppose we now read the second register, and get (for example) 2. That means the first register goes into the state $(1/\sqrt{86})\big[|1\rangle + |7\rangle + |13\rangle + \ldots + |505\rangle + |511\rangle\big] =$ $(1/\sqrt{86})\sum_{i=0}^{84} |6i + 1\rangle$. (We hope to find the 6 = r.) Shor then applies a Fourier transform, and ends with a state having the property that when read, has a good chance of producing a number t such that t/512 is a good approximation to a number of the form k/r. The theory of continued fractions allows us to find r, (with luck, the right r). We apply exponential factorization with r. If we fail to factor 21, we know we got the wrong r. With luck, a few tries gives us the right r, namely 6.

Remark: It has been said that Shor's algorithm is a special case of the hidden subgroup problem. Although there are similarities, perhaps it would be better to say both are examples of hidden cycle problems.

In [K], Kiteav introduces a new idea, which gives an alternate factorization algorithm, with some similarities to Shor's. (See [M3].) However, there are aspects of [K] that I do not yet understand.

REFERENCES

[I] I. M. Isaacs, Character theory of finite groups, Dover Publications, inc.

[K] A. Y. Kiteav, Quantum measurements and the Abelian stabilizer problem, arXiv:quant-ph/9511026v1, November 20, 1995

[L] C. Lomont, The hidden subgroup problem—review and open problems, raXiv:quant-ph/0411037v1, November 4, 2004

[M1] S. McAdam, Shor's algorithms, www.ma.utexas.edu/users/mcadam

[M2] S. McAdam, Entanglement, www.ma.utexas.edu/users/mcadam

[M3] S. McAdam, Eigenvalues and Kiteav's factoring algorithm, www.ma.utexas.edu/users/mcadam

[M4] S. McAdam, Unkonowable Matters, American Mathematical Monthly, 119 (4) 2012, 284-289

[S] P. Shor, Polynomial time algorithms for prime factorization and discrete logarithms on a quantum computer, available at xxx.lanl.gov/abs/quant-ph/9508027