Quantum algorithms (CO 781/CS 867/QIC 823, Winter 2013)
Andrew Childs, University of Waterloo

LECTURE 3: The abelian hidden subgroup problem

In this lecture, we will introduce the general hidden subgroup problem (HSP). We'll see how Shor's discrete log algorithm solves a particular instance of the HSP in an Abelian group. Finally, we'll see how to solve the HSP in any finite abelian group of known structure.

The hidden subgroup problem

In the general HSP, we are given a black-box function $f\colon G \to S$, where G is a known group and S is a finite set. The function is promised to satisfy

$$f(x) = f(y) \text{ if and only if } x^{-1}y \in H \quad \text{i.e., } y = xh \text{ for some } h \in H \tag{1}$$

for some unknown subgroup $H \leq G$. We say that such a function *hides* H. The goal of the HSP is to learn H (say, specified in terms of a generating set) using queries to f.

It's clear that H can in principle be reconstructed if we are given the entire truth table of f. Notice in particular that $f(1) = f(x)$ if and only if $x \in H$: the hiding function is constant on the hidden subgroup, and does not take that value anywhere else.

But the hiding function has a lot more structure than this. If we fix some element $g \in G$ with $g \notin H$, we see that $f(g) = f(x)$ if and only if $x \in gH$, a left coset of H in G with coset representative g. So f is constant on the left cosets of H in G, and distinct on different left cosets.

In the above definition of the HSP, we have made an arbitrary choice to multiply by elements of H on the right, which is why the hiding function is constant on left cosets. We could just as well have chosen to multiply by elements of H on the left, in which case the hiding function would be constant on right cosets; the resulting problem would be equivalent. Of course, in the case where G is abelian, we don't need to make such a choice. For reasons that we will see later, this case turns out to be considerably simpler than the general case; indeed, there is an efficient quantum algorithm for the HSP in any abelian group, whereas there are only a few nonabelian groups for which efficient algorithms are known.

You should be familiar with Simon's problem, which is simply the HSP with $G = \mathbb{Z}_2^n$ and $H = \{0, s\}$ for some $s \in \mathbb{Z}_2^n$. There is a straightforward quantum algorithm for this problem, yet one can prove that any classical algorithm for finding s must query the hiding function exponentially many times (in n). The gist of the argument is that, since the set S is unstructured, we can do no better than querying random group elements so long as we do not know two elements x, y for which $f(x) = f(y)$. But by the birthday problem, we are unlikely to see such a collision until we make $\Omega(\sqrt{|G|/|H|})$ random queries.

A similar argument applies to any HSP with a large number of trivially intersecting subgroups. More precisely, we have

Theorem. *Suppose that G has a set $\mathcal{H}$ of N subgroups whose only common element is the identity. Then a classical computer must make $\Omega(\sqrt{N})$ queries to solve the HSP.*

Proof. Suppose the oracle does not a priori hide a particular subgroup, but instead behaves adversarially, as follows. On the ℓth query, the algorithm queries g_ℓ, which we assume to be different

from $g_1, \ldots, g_{\ell-1}$ without loss of generality. If there is any subgroup $H \in \mathcal{H}$ for which $g_k \notin g_j H$ for all $1 \leq j < k \leq \ell$ (i.e., there is some consistent way the oracle could assign g_ℓ to an as-yet-unqueried coset of a hidden subgroup from $\mathcal{H}$), then the oracle simply outputs ℓ; otherwise the oracle conceeds defeat and outputs a generating set for some $H \in \mathcal{H}$ consistent with its answers so far (which must exist, by construction).

The goal of the algorithm is to force the oracle to conceed, and we want to lower bound the number of queries required. (Given an algorithm for the HSP in G, there is clearly an algorithm that forces this oracle to conceed using only one more query.) Now consider an algorithm that queries the oracle t times before forcing the oracle to conceed. This algorithm simply sees a fixed sequence of responses $1, 2, \ldots, t$, so for the first t queries, the algorithm cannot be adaptive. But observe that, regardless of which t group elements are queried, there are at most $\binom{t}{2}$ values of $g_k g_j^{-1}$, whereas there are N possible subgroups in $\mathcal{H}$. Thus, to satisfy the N conditions that for all $H \in \mathcal{H}$, there is some pair j, k such that $g_k g_j^{-1} \in H$, we must have $\binom{t}{2} \geq N$, i.e., $t = \Omega(\sqrt{N})$. $\square$

Note that there are cases where a classical algorithm *can* find the hidden subgroup with a polynomial number of queries. In particular, since a classical computer can easily test whether a certain subgroup is indeed the hidden one, the HSP is easy for a group with only polynomially many subgroups. For example, a classical computer can easily solve the HSP in $\mathbb{Z}_p$ for p prime (since it has only 2 subgroups) and in $\mathbb{Z}_{2^n}$ (since it has only $n+1$ subgroups).

Discrete log as a hidden subgroup problem

The discrete log problem is easily recognized as an HSP. Recall that Shor's algorithm for computing $\log_g x$ involves the function $f\colon \mathbb{Z}_N \times \mathbb{Z}_N \to \langle g \rangle$ defined by $f(\alpha, \beta) = x^\alpha g^\beta$. This function is constant on the lines $L_\gamma = \{(\alpha, \beta) \in \mathbb{Z}_N^2 \colon \alpha \log_g x + \beta = \gamma\}$. Observe that $H = L_0$ is a subgroup of $G = \mathbb{Z}_N \times \mathbb{Z}_N$, and the sets $L_\gamma = L_0 + (0, \gamma)$ are its cosets. Shor's algorithm for discrete log works by making the coset state $|L_\gamma\rangle$ for a uniformly random γ and measuring in the Fourier basis.

The abelian HSP

We now consider the HSP for a general abelian group. When the group elements commute, it often makes more sense to use additive notation for the group operation. We use this convention here, writing the condition that f hides H as $f(x) = f(x)$ iff $x - y \in H$.

The strategy for the general abelian HSP closely follows the algorithm for the discrete log problem. We begin by creating a uniform superposition over the group,

$$|G\rangle := \frac{1}{\sqrt{|G|}} \sum_{x \in G} |x\rangle. \tag{2}$$

Then we compute the function value in another register, giving

$$\frac{1}{\sqrt{|G|}} \sum_{x \in G} |x, f(x)\rangle. \tag{3}$$

Discarding the second register then gives a uniform superposition over the elements of some randomly chosen coset $x + H := \{x + h \colon h \in H\}$ of H in G,

$$|x + H\rangle = \frac{1}{\sqrt{|H|}} \sum_{h \in H} |x + h\rangle. \tag{4}$$

Such a state is commonly called a *coset state*. Equivalently, since the coset is unknown and uniformly random, the state can be described by the density matrix

$$\rho_H := \frac{1}{|G|} \sum_{x \in G} |x+H\rangle\langle x+H|. \tag{5}$$

Next we apply the QFT over G. Then we obtain the state

$$\begin{aligned} |\widehat{x+H}\rangle &:= F_G |x+H\rangle & (6)\\ &= \frac{1}{\sqrt{|H| \cdot |G|}} \sum_{y \in \hat{G}} \sum_{h \in H} \chi_y(x+h)|y\rangle & (7)\\ &= \sqrt{\frac{|H|}{|G|}} \sum_{y \in \hat{G}} \chi_y(x)\chi_y(H)|y\rangle & (8) \end{aligned}$$

where

$$\chi_y(H) := \frac{1}{|H|} \sum_{h \in H} \chi_y(h). \tag{9}$$

Note that applying the QFT was the right thing to do because the state ρ_H is G-invariant. In other words, it commutes with the regular representation of G, the unitary matrices $U(x)$ satisfying $U(x)|y\rangle = |x+y\rangle$ for all $x, y \in G$: we have

$$\begin{aligned} U(x)\rho_H &= \frac{1}{|G|} \sum_{y \in G} |x+y+H\rangle\langle y+H| & (10)\\ &= \frac{1}{|G|} \sum_{z \in G} |z+H\rangle\langle z-x+H| & (11)\\ &= \rho_H U(-x)^\dagger & (12)\\ &= \rho_H U(x). & (13) \end{aligned}$$

It follows that $\hat{\rho}_H := F_G \rho_H F_G^\dagger$ is diagonal (indeed, we verify this explicitly below), so we can measure without losing any information. We will talk about this phenomenon more when we discuss nonabelian Fourier sampling.

Note that χ_y is a character of H if we restrict our attention to that subgroup. If $\chi_y(h) = 1$ for all $h \in H$, then clearly $\chi_y(H) = 1$. On the other hand, if there is any $h \in H$ with $\chi_y(h) \neq 1$ (i.e., if the restriction of χ_y to H is not the trivial character of H), then by the orthogonality of distinct characters,

$$\frac{1}{|H|} \sum_{x \in H} \chi_y(x)\chi_{y'}(x)^* = \delta_{y,y'} \tag{14}$$

(equivalent to unitarity of the QFT), we have $\chi_y(H) = 0$. Thus we have

$$|\widehat{x+H}\rangle = \sqrt{\frac{|H|}{|G|}} \sum_{y:\ \chi_y(H)=1} \chi_y(x)|y\rangle \tag{15}$$

or, equivalently, the mixed quantum state

$$\hat{\rho}_H = \frac{|H|}{|G|^2} \sum_{x \in G} \sum_{y,y' \colon \chi_y(H) = \chi_{y'}(H) = 1} \chi_y(x) \chi_{y'}(x) |y\rangle\langle y'| = \frac{|H|}{|G|} \sum_{y \colon \chi_y(H) = 1} |y\rangle\langle y|. \tag{16}$$

Next we measure in the computational basis. Then we obtain some character χ_y that is trivial on the hidden subgroup H. This information narrows down the possible elements of the hidden subgroup: we can restrict our attention to those elements $g \in G$ satisfying $\chi_y(g) = 1$. The set of such elements is called the *kernel* of χ_y,

$$\ker \chi_y := \{g \in G \colon \chi_y(g) = 1\}; \tag{17}$$

it is a subgroup of G. Now our strategy is to repeat the entire sampling procedure many times and compute the intersection of the kernels of the resulting characters. After only polynomially many steps, we claim that the resulting subgroup is H with high probability. It clearly cannot be smaller than H (since the kernel of every sampled character contains H), so it suffices to show that each sample is likely to reduce the size of H by a substantial fraction until H is reached.

Suppose that at some point in this process, the intersection of the kernels is $K \leq G$ with $K \neq H$. Since K is a subgroup of G with $H < K$, we have $|K| \geq 2|H|$ (by Lagrange's theorem). Because each character χ_y of G satisfying $\chi_y(H)$ has probability $|H|/|G|$ of appearing, the probability that we see some χ_y for which $K \leq \ker \chi_y$ is

$$\frac{|H|}{|G|} |\{y \in \hat{G} \colon K \leq \ker \chi_y\}|. \tag{18}$$

But the number of such ys is precisely $|G|/|K|$, since we know that if the subgroup K were hidden, we would sample such ys uniformly, with probability $|K|/|G|$. Therefore the probability that we see a y for which $K \leq \ker \chi_y$ is precisely $|H|/|K| \leq 1/2$. Now if we observe a y such that $K \not\leq \ker \chi_y$, then $|K \cap \ker \chi_y| \leq |K|/2$; furthermore, this happens with probability at least $1/2$. Thus, if we repeat the process $O(\log |G|)$ times, it is extremely likely that the resulting subgroup is in fact H.

Decomposing abelian groups

To apply the above algorithm, we must understand the structure of the group G; in particular, we must be able to apply the Fourier transform F_G. For some applications, we might not know the structure of G a priori. But if we assume only that we have a unique encoding of each element of G, the ability to perform group operations on these elements, and a generating set for G, then there is an efficient quantum algorithm (due to Mosca) that decomposes the group as

$$G = \langle \gamma_1 \rangle \oplus \langle \gamma_2 \rangle \oplus \cdots \oplus \langle \gamma_t \rangle \tag{19}$$

in terms of generators $\gamma_1, \gamma_2, \ldots, \gamma_t$. Here $\oplus$ denotes an internal direct sum, meaning that the groups $\langle \gamma_i \rangle$ intersect only in the identity element; in other words, we have

$$G \cong \mathbb{Z}_{|\langle \gamma_1 \rangle|} \times \mathbb{Z}_{|\langle \gamma_2 \rangle|} \times \cdots \times \mathbb{Z}_{|\langle \gamma_t \rangle|}. \tag{20}$$

Given such a decomposition, it is straightforward to implement F_G and thereby solve HSPs in G. We might also use this tool to decompose the structure of the hidden subgroup H output by the HSP algorithm, e.g., to compute $|H|$.

This algorithm is based on Shor's algorithm for order finding, together with standard tools from group theory. We will not have time to cover the algorithm in detail; for more, see the lecture notes from 2011.