Hidden Subgroup Problem

Overview

Simon's problem and period finding have essentially the same quantum algorithm:

- Create the uniform superposition
- Query the function
- Fourier transform over appropriate group
- Classical post-processing

Today we will see they are a common instance of the hidden subgroup problem.

Review

Simon:

Promise: Given $f : \mathbb{Z}_2^n \rightarrow [M]$ there is an s such that $f(x) = f(y)$ iff $x - y \in \{0^n, s\}$.

Goal: Find s.

(Simple) period finding:

Promise: Given $f : \mathbb{Z}_N \rightarrow [M]$ there is an s such that $f(x) = f(y)$ iff $x - y = 0 \bmod s$.

Goal: Find s.

Group

To identify the commonality we need a bit of group theory.

A group $(G, \circ)$ is a set together with a binary operation $\circ : G \times G \to G$ that satisfy 3 conditions:

1) Associativity: For all $x, y, z \in G$

$$(x \circ y) \circ z = x \circ (y \circ z)$$

2) Identity: There exists $e \in G$ s.t. for all $x \in G$

$$e \circ x = x \circ e = x$$

3) Inverses: For all $x \in G$ there is a $y \in G$ such that

$$x \circ y = y \circ x = e$$

Subgroup

Let $(G, \circ)$ be a group. A subgroup $H \leq G$ is a subset H satisfying the three conditions:

1) Closure: $x \circ y \in H$ for all $x, y \in H$.

2) Identity: $e \in H$

2) Closed under inverses: $x^{-1} \in H$ for all $x \in H$

Example

$\mathbb{Z}_N$ is a group under addition modulo N .

The identity element is 0.

If $s|N$ then $H = \{0, s, 2s, \ldots, N-s\}$ is a subgroup.

In the (simple) period finding we are in particular promised that f is constant on H .

Where else is f constant?

Cosets

$H = \{0, s, 2s, \ldots, N - s\}$ is a subgroup of $\mathbb{Z}_N$.

f is also constant on the set

$$\begin{aligned} H_1 &= \{1, s+1, 2s+1, \ldots, N-s+1\} \\ &= \{x \in \mathbb{Z}_N : x = 1 \bmod s\} \end{aligned}$$

This set is not itself a subgroup.

$$H_1 = \{1 + h : h \in H\} = 1 + H$$

This is a coset!

Cosets

Let $(G, \circ)$ be a group and $H \leq G$ a subgroup.

Every $g \in G$ defines a (left) coset of H

$$\{g \circ h : h \in H\}$$

These sets are either identical or disjoint.

Distinct cosets form a partition of G .

One can also define right cosets. For abelian groups I'll just call them cosets.

Period finding

Let's go back to $\mathbb{Z}_N$ and the subgroup

$$H = \{0, s, 2s, \ldots, N - s\}$$

Another way to state the promise on the function f is

1) f is constant on cosets of H.

2) Distinct cosets take on distinct values under f.

Goal: The goal of finding s is equivalent to learning the subgroup H.

Simon's problem

For Simon's problem we work over the group $\mathbb{Z}_2^n$.

The operation is bitwise addition modulo 2.

The identity element is 0^n.

Each element is its own inverse.

For Simon's problem the relevant subgroup is $\{0^n, s\}$.

Simon's problem

Promise: Given $f : \mathbb{Z}_2^n \to [M]$ there is an s such that $f(x) = f(y)$ iff $x - y \in \{0^n, s\}$.

Let $H = \{0^n, s\}$. We can reformulate the promise as

1) f is constant on cosets of H.

2) Distinct cosets take on distinct values under f .

Goal: The goal of finding s is equivalent to learning the subgroup H .

Hidden Subgroup Problem

Now we see a common generalization of these problems.

Let G be a group and $H \leq G$.

Promise: Let f be a function such that

1) f is constant on left cosets of H.

2) Distinct cosets take on distinct values under f.

Goal: Find H.

Remarks

We assume oracle access to f.

What does it mean to "find H"?

Find a generating set for H, a set of elements T whose closure $\langle T \rangle$ under the group operation and taking inverses is H.

There is always a generating set of size $\log(|H|)$.

Discrete Logarithm

Discrete Logarithm

Now let's see a new problem that fits into this framework.

In the discrete logarithm problem one considers a cyclic group G generated by an element g.

That is, $G = \{1, g, g^2, \ldots, g^{N-1}\}$ and $g^N = 1$.

Discrete log problem: Given $x \in G$ find the least non-negative a such that $x = g^a$.

We denote this as $a = \log_g(x)$.

Example

Let's look at the operation of multiplication modulo 11.

Look at the group generated by 7.

$$7^1 = 7 \bmod 11 \qquad 7^6 = 4 \bmod 11$$
$$7^2 = 5 \bmod 11 \qquad 7^7 = 6 \bmod 11$$
$$7^3 = 2 \bmod 11 \qquad 7^8 = 9 \bmod 11$$
$$7^4 = 3 \bmod 11 \qquad 7^9 = 8 \bmod 11$$
$$7^5 = 10 \bmod 11 \qquad 7^{10} = 1 \bmod 11$$

This is the multiplicative group modulo 11 denoted $\mathbb{Z}_{11}^{\times}$.

Example

$$7^1 = 7 \bmod 11 \qquad 7^6 = 4 \bmod 11$$

$$7^2 = 5 \bmod 11 \qquad 7^7 = 6 \bmod 11$$

$$7^3 = 2 \bmod 11 \qquad 7^8 = 9 \bmod 11$$

$$7^4 = 3 \bmod 11 \qquad 7^9 = 8 \bmod 11$$

$$7^5 = 10 \bmod 11 \qquad 7^{10} = 1 \bmod 11$$

In $\mathbb{Z}_{11}^{\times}$:

$$\log_7(10) = 5$$

$$\log_7(6) = 7$$

$$\log_7(1) = ?$$

Example 2

In $\mathbb{Z}_{163}^{\times}$ what is $\log_{18}(65)$?

$$\log_{18}(65) = 132$$

$$18^{132} = 18^{128+4}$$

Check:

exponent	value	mod163
1	18	18
2	18^2	161
4	161^2	4
8	4^2	16
16	16^2	93
32	93^2	10
64	10^2	100
128	100^2	57

$$18^{132} \bmod 163$$

$$= (18^{128} \bmod 163) \cdot (18^4 \bmod 163)$$

$$= 57 \cdot 4 \bmod 163 = 65$$

Example 2

In $\mathbb{Z}_{163}^{\times}$, $\log_{18}(65) = 132$.

Hopefully this example shows that it is easy to compute

$$18^{132} \bmod 163$$

but seems hard to compute $\log_{18}(65)$.

Classically, the best heuristic algorithms over $\mathbb{Z}_p^{\times}$ take time roughly $\exp(\log(p)^{1/3})$.

Discrete log and factoring

There is a close connection between discrete log and factoring.

The ability to solve discrete logs in $\mathbb{Z}_N^\times$ lets you factor N [Bach 84, Miller 76].

You can use discrete log to compute the order of a random $x \in \mathbb{Z}_N^\times$.

To solve discrete logs in $\mathbb{Z}_N^\times$ it suffices to factor N and be able to solve discrete logs in $\mathbb{Z}_p^\times$ for primes p [Bach 84].

Discrete log and factoring

The ability to solve discrete logs in $\mathbb{Z}_N^\times$ lets you factor N [Bach 84, Miller 76].

The analysis of Shor's discrete log algorithm is much simpler than that of his factoring algorithm.

However, Shor's discrete log algorithm requires knowing the order of the group.

In the case of $\mathbb{Z}_N^\times$ this is $\phi(N)$ and computing this is as hard as factoring.

Diffie Helman key exchange

Discrete log is the basis of a beautiful protocol for two parties to share a secret key.

Alice and Bob agree on a prime modulus p and and generator g for $\mathbb{Z}_p^\times$.

$$p = 163, g = 2$$

Alice randomly chooses some $1 < a < p - 1$.

Bob randomly chooses some $1 < b < p - 1$.

Diffie Helman key exchange

Alice randomly chooses some $1 < a < p - 1$.

Bob randomly chooses some $1 < b < p - 1$.

Alice sends $A = g^a \bmod p$ to Bob and Bob sends $B = g^b \bmod p$ to Alice.

Alice computes $B^a \bmod p$ and Bob computes $A^b \bmod p$.

This is the shared secret!

$$B^a \bmod p = g^{ab} \bmod p = A^b \bmod p$$

Diffie Helman key exchange

What is publicly known is $g, p, g^a \bmod p, g^b \bmod p$

If you could compute discrete logs, then you could also compute the shared secret $g^{ab} \bmod p$.

This is still the best known way to compute the secret from the publicly known information.

On the other hand, there is no proof that breaking Diffie-Helman is as hard as computing discrete logs.

Discrete Log and HSP

Let $G = \langle g \rangle$ with $|G| = N$.

Given $x \in G$ we want to compute $\log_g(x)$.

We can formulate this as a HSP in the (additive) group $\mathbb{Z}_N \times \mathbb{Z}_N$.

$\mathbb{Z}_N \times \mathbb{Z}_N = \mathbb{Z}_N^2$

$$\{(a,b) : a, b \in \mathbb{Z}_N\}$$

$$(a,b) + (c,d) = (a + c \bmod N, b + d \bmod N)$$

Let $f : \mathbb{Z}_N \times \mathbb{Z}_N \to G$ be defined as $f(a,b) = x^a g^b$.

Discrete Log and HSP

Let $f : \mathbb{Z}_N \times \mathbb{Z}_N \to G$ be defined as $f(a,b) = x^a g^b$.

We can write this as $f(a,b) = g^{a \log_g(x) + b}$.

So f is constant on the lines $g^N = 1 \bmod N$

$$L_c = \{(a,b) : a \log_g(x) + b = c \bmod N\}$$

Also $g^c \neq g^{c'}$ for $1 < c \neq c' < N$ so f takes distinct values on different L_c .

$\mathrm{ord}(g) \leq N$

$$L_c = \{(a,b) : a \log_g(x) + b = c \bmod N\}$$

It just remains to show these are cosets of a subgroup H.

$$H = L_0 = \{(0,0), (1, -\log_g(x)), (2, -2\log_g(x)), \ldots, (N-1, -(N-1)\log_g(x))\}$$

- It contains the identity.
- It is closed under the group operation:

$$(k_1, -k_1 \log_g(x)) + (k_2, -k_2 \log_g(x)) = (k_1 + k_2, -(k_1 + k_2)\log_g(x))$$

- It is closed under inverses:

$$(-k, k\log_g(x)) \in H \quad \text{for} \quad k \in \mathbb{Z}_N$$

$$L_c = \{(a,b) : a \log_g(x) + b = c \bmod N\}$$

$$H = L_0 = \{(0,0), (1, -\log_g(x)), (2, -2\log_g(x)), \ldots, (N-1, -(N-1)\log_g(x))\}$$

$$L_c = \{(0,c), (1, -\log_g(x) + c), (2, -2\log_g(x) + c), \ldots, (N-1, -(N-1)\log_g(x) + c)\}$$

$$= (0,c) + H$$

Each L_c is a coset of H, and these form a complete set of cosets.

Thus discrete logarithm is an instance of the HSP.

Note that $|H| = N$.

Quantum Alg. for Discrete Log

Q. Alg. for discrete log

The quantum algorithm for discrete log is very similar to what we have already seen.

What do you guess the first two steps are?

Problem: Let $G = \langle g \rangle$ with $|G| = N$.

Given $h \in G$ we want to compute $\log_g(h)$.

Recall: $f : \mathbb{Z}_N \times \mathbb{Z}_N \to G$ with $f(a,b) = h^a g^b$.

Step 1: Prepare the uniform superposition over $\mathbb{Z}_N^2$.

$$|0\rangle|0\rangle|0\rangle \mapsto \frac{1}{N} \sum_{a,b \in \mathbb{Z}_N} |a\rangle|b\rangle|0\rangle$$

Step 2: Apply f .

$$\frac{1}{N} \sum_{a,b \in \mathbb{Z}_N} |a\rangle|b\rangle|0\rangle \mapsto \frac{1}{N} \sum_{a,b \in \mathbb{Z}_N} |a\rangle|b\rangle|f(a,b)\rangle$$

$$\frac{1}{N}\sum_{a,b\in\mathbb{Z}_N}|a\rangle|b\rangle|f(a,b)\rangle$$

Now we measure the second register.

For a uniformly random $c\in\mathbb{Z}_N$ we are in the state $|L_c\rangle|g^c\rangle$ where $|L_c\rangle$ is the coset state

$$|L_c\rangle = \frac{1}{\sqrt{N}}\sum_{(a,b)\in L_c}|a\rangle|b\rangle$$

$$= \frac{1}{\sqrt{N}}\sum_{(a,b)\in H}|a\rangle|b+c\rangle$$

where recall $L_c = \{(a,b) : a\log_g(h)+b = c \bmod N\}$ and $H = L_0$.

Fourier Transform

Next we apply the Fourier Transform over $\mathbb{Z}_N \times \mathbb{Z}_N$ to the first two registers.

Let F_N be the Fourier transform over $\mathbb{Z}_N$.

$$F_N|x\rangle = \frac{1}{\sqrt{N}} \sum_{y \in \mathbb{Z}_N} \omega^{-xy} |y\rangle \qquad x \in \mathbb{Z}_N$$

The Fourier transform over $\mathbb{Z}_N \times \mathbb{Z}_N$ is $F_N \otimes F_N$.

Punchline

$\omega = \exp(2\pi \mathrm{i}/N)$

$$(F_N \otimes F_N)|L_c\rangle = \frac{1}{\sqrt{N}} \sum_{(a,b)\in S} \omega^{-bc}|a\rangle|b\rangle$$

$$S = \{(a,b) : a - b\log_g(h) = 0 \bmod N\}$$

$$= \{(b\log_g(h) \bmod N, b) : b \in \mathbb{Z}_N\}$$

Again the zero/nonzero pattern doesn't depend on c.

Independently of c, when we measure we see

$$(b\log_g(h) \bmod N, b)$$

for a uniformly random $b \in \mathbb{Z}_N$.

Finishing the algorithm

Independently of c, when we measure we see

$$(b \log_g(h) \bmod N, b)$$

for a uniformly random $b \in \mathbb{Z}_N$.

If $\gcd(b, N) = 1$ then we can multiply by $b^{-1} \in \mathbb{Z}_N^\times$ and recover $\log_g(h)$.

The probability that $\gcd(b, N) = 1$ is at least $\dfrac{1}{4 \log \log N}$ for $N \geq 16$.

Summary

To succeed with high probability we repeat the quantum procedure $100 \log \log N$.

Each quantum procedure consists of two applications of the Fourier transform and one query to f.

Classical post-processing is $O(\log N)$ time.

Total quantum gate complexity is $O(\log^2(N) \log \log N)$ and number of queries is $O(\log \log(N))$.

Lab

Try different values of x, g, M.

Program finds $\log_g(x)$ in $\mathbb{Z}_M^\times$ if it exists.

1) What is N in this case?

X is an N-by-N matrix such that $(j, X(j,c)) \in L_c$

2) verify that $f(a,b) = x^a g^b$ is constant on L_c.

3) Use Yhat to check the formula for $(F_N \otimes F_n)|L_c\rangle$.

4) Why does the program "work" even when N is wrong?

Proof of Punchline

To prove: $$(F_N \otimes F_N)|L_c\rangle = \frac{1}{\sqrt{N}} \sum_{(a,b)\in S} \omega^{-bc}|a\rangle|b\rangle$$

$$S = \{(a,b) : a - b\log_g(h) = 0 \bmod N\}$$

Recall: $$|L_c\rangle = \frac{1}{\sqrt{N}} \sum_{(x,y)\in H} |x\rangle|y+c\rangle$$

$$H = \{(k, -k\log_g(h) \bmod N) : k \in \mathbb{Z}\}$$

Proof of Punchline

To prove: $$(F_N \otimes F_N)|L_c\rangle = \frac{1}{\sqrt{N}} \sum_{(a,b)\in S} \omega^{-bc}|a\rangle|b\rangle$$

$$S = \{(a,b) : a - b\log_g(h) = 0 \bmod N\}$$

$$(F_N \otimes F_N)\frac{1}{\sqrt{N}} \sum_{(x,y)\in H} |x\rangle|y+c\rangle = \frac{1}{N^{3/2}} \sum_{(x,y)\in H} \sum_{a,b\in\mathbb{Z}_N} \omega^{-ax-b(y+c)}|a\rangle|b\rangle$$

$$= \frac{1}{N^{3/2}} \sum_{a,b\in\mathbb{Z}_N} \omega^{-bc}|a\rangle|b\rangle \sum_{(x,y)\in H} \omega^{-ax-by}$$

Now focus on this term

Sum over subgroup

$$\sum_{(x,y)\in H} \omega^{-ax-by}$$

If there is an $(x', y') \in H$ with $\omega^{-ax'-by'} \neq 1$ then this sum is zero.

The reason is as we saw in the period finding problem:

$$(x', y') + H = H$$

$$\sum_{(x,y)\in H} \omega^{-ax-by} = \sum_{(x,y)\in H} \omega^{-a(x+x')-b(y+y')} = \omega^{-ax'-by'} \sum_{(x,y)\in H} \omega^{-ax-by}$$

Sum over subgroup

$$\sum_{(x,y)\in H} \omega^{-ax-by}$$

When is $\omega^{-ax'-by'} = 1$ for all $(x', y') \in H$?

This happens iff $a - b\log_g(h) = 0 \bmod N$.

$$\sum_{(x,y)\in H} \omega^{-ax-by} = \begin{cases} N & \text{if } a - b\log_g(h) = 0 \bmod N \\ 0 & \text{otherwise} \end{cases}$$

Proof of Punchline

Let's go back to our expression:

$$(F_N \otimes F_N)|L_c\rangle = \frac{1}{N^{3/2}} \sum_{a,b \in \mathbb{Z}_N} \omega^{-bc}|a\rangle|b\rangle \sum_{(x,y)\in H} \omega^{-ax-by}$$

$$= \frac{1}{\sqrt{N}} \sum_{a,b \in S} \omega^{-bc}|a\rangle|b\rangle$$

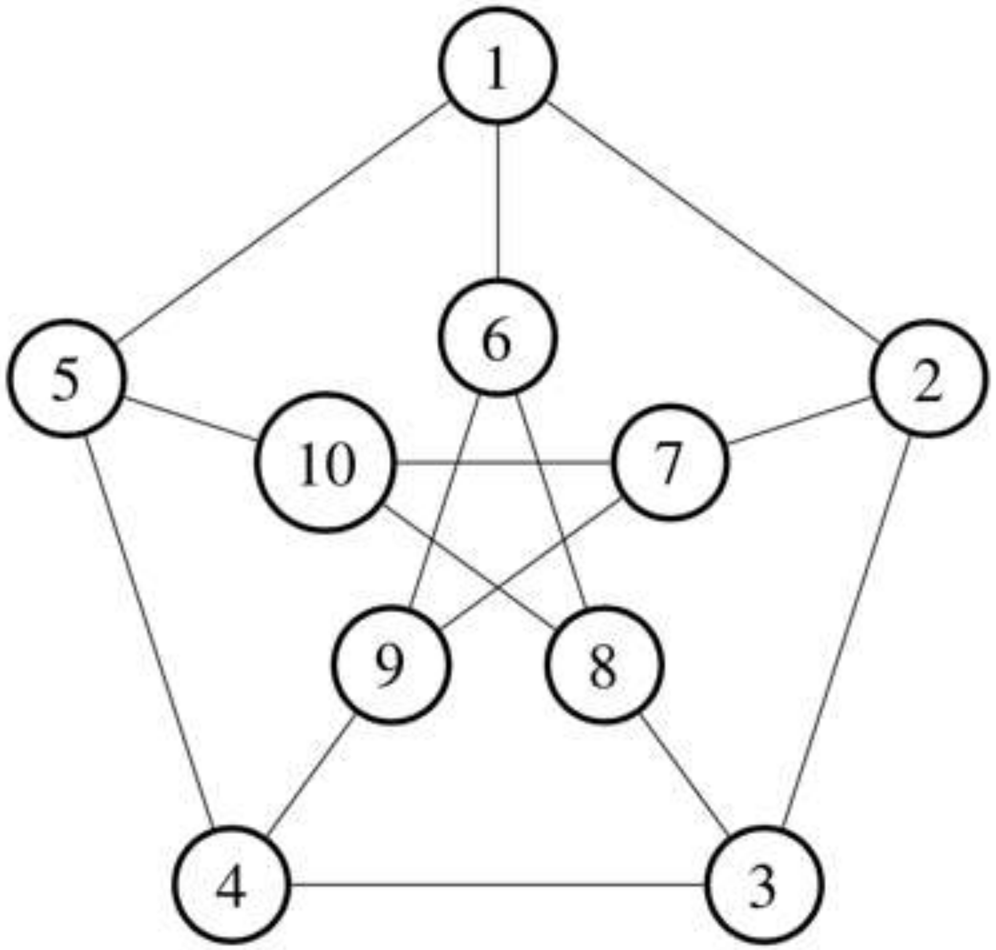
$$S = \{(a,b) : a - b\log_g(h) = 0 \bmod N\}$$

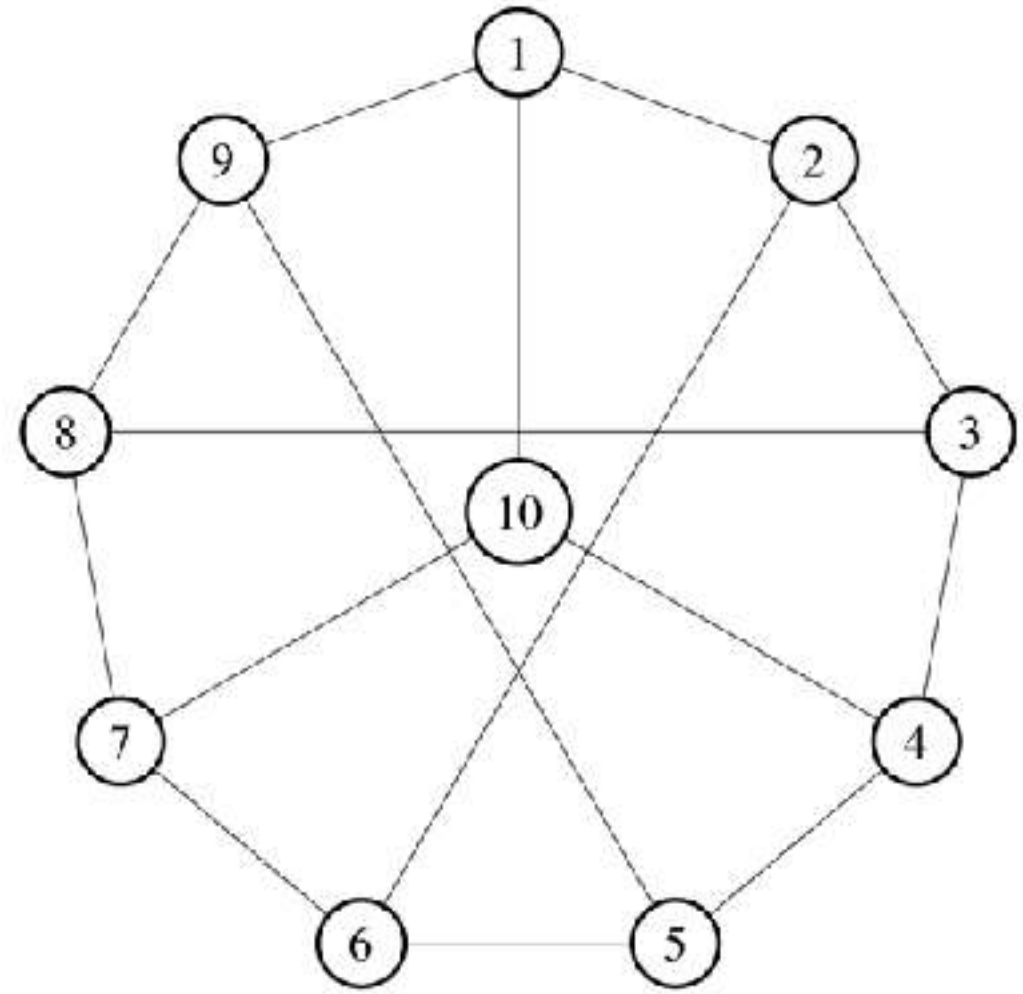
As we claimed!

Non-abelian HSP

Graph Isomorphism

Are these two graphs "the same"?





They both have 10 vertices.

They both have 15 edges.

Graph Isomorphism

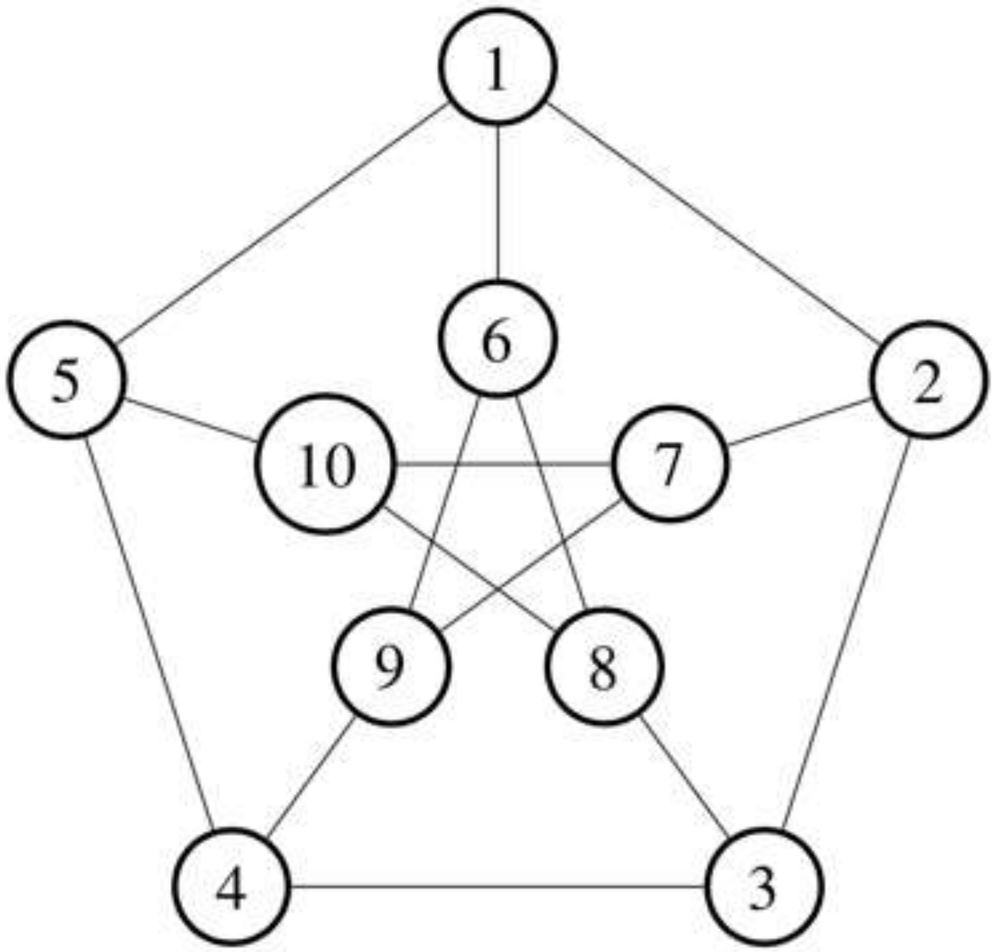
What should we mean by the same?

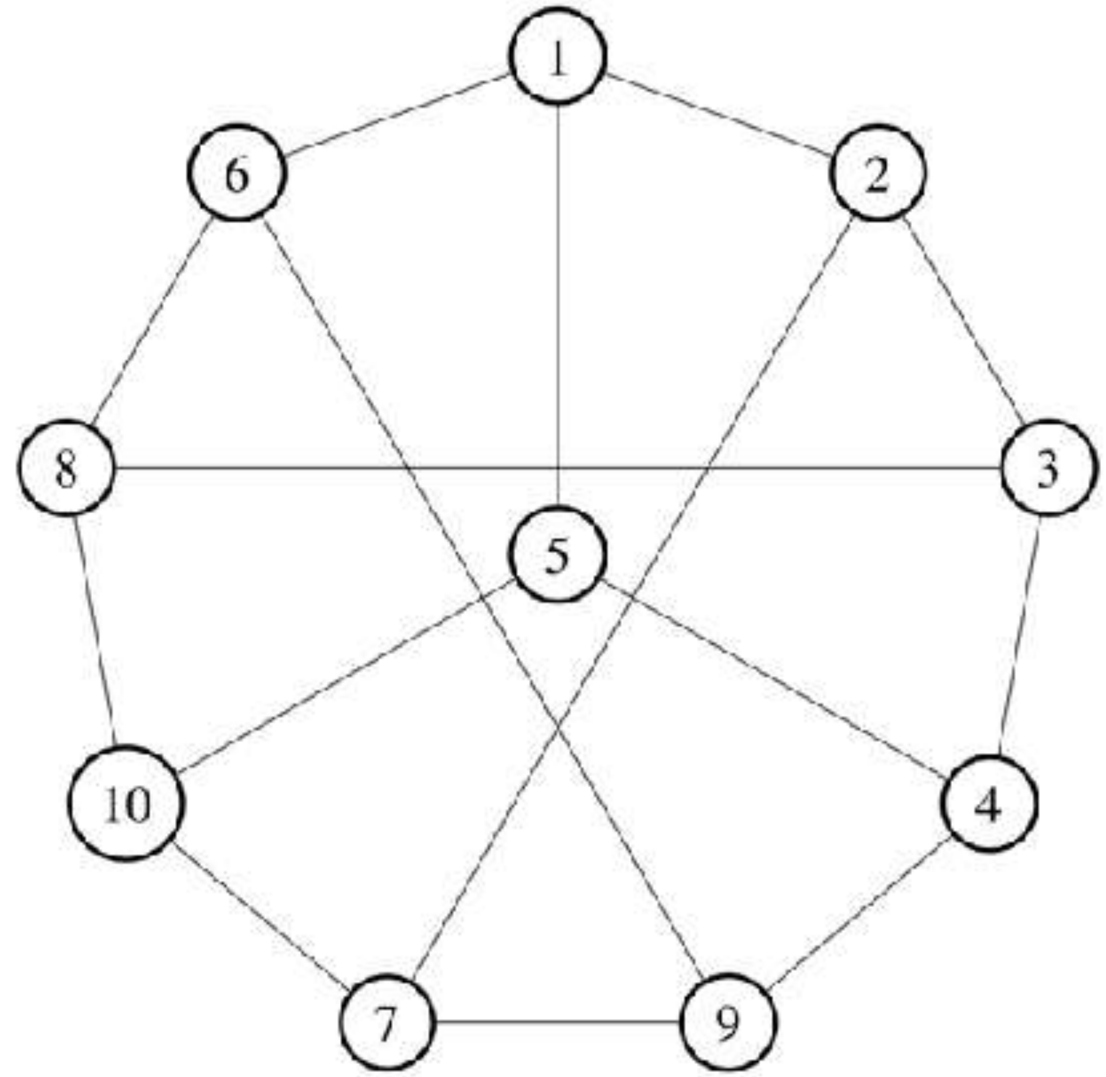
Definition: Graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are isomorphic if there is a bijection $\pi : V_1 \to V_2$ s.t.

$$(x, y) \in E_1 \iff (\pi(x), \pi(y)) \in E_2$$

Graph Isomorphism

These graphs are isomorphic!





Complexity

Graph isomorphism is in NP.

It is not thought to be NP-complete and it is also not known to be in P.

Babai has given a quasi-polynomial time classical algorithm.

We next show that graph isomorphism is a special case of the non-abelian HSP.

Graph Automorphism

We start with an easier reduction to the HSP.

An automorphism of an n-vertex graph $\mathcal{G} = (V, E)$ is a bijection $\pi : V \to V$ such that

$$(x, y) \in E \iff (\pi(x), \pi(y)) \in E$$

The set of all automorphisms $\mathrm{Aut}(\mathcal{G})$ is a subgroup of S_n.

The graph automorphism problem is to determine if $\mathcal{G}$ has a non-trivial automorphism.

Graph Automorphism

An automorphism of an n-vertex graph $\mathcal{G} = (V, E)$ is a bijection $\pi : V \to V$ such that

$$(x, y) \in E \iff (\pi(x), \pi(y)) \in E$$

The function $f(\sigma) = \sigma(\mathcal{G})$ for $\sigma \in S_n$ hides $\mathrm{Aut}(\mathcal{G})$.

If $\sigma \in \mathrm{Aut}(G)$ then $\sigma_1\sigma(G) = \sigma_1(G)$. Constant on cosets.

If $\sigma_1(G) = \sigma_2(G)$ then $\sigma_2^{-1}\sigma_1 \in \mathrm{Aut}(G)$. In same coset.

Thus an algorithm to solve HSP over S_n can also solve the graph automorphism problem.

Graph Isomorphism

Say that we want to tell if the graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are isomorphic.

High level: we find the automorphism group of the graph $\mathcal{G} = (V_1 \sqcup V_2, E_1 \sqcup E_2)$.

$\mathcal{G}_1, \mathcal{G}_2$ are isomorphic iff there is a permutation $\pi \in \mathrm{Aut}(\mathcal{G})$ that exchanges V_1 and V_2.

This can be formalized as a HSP in the group

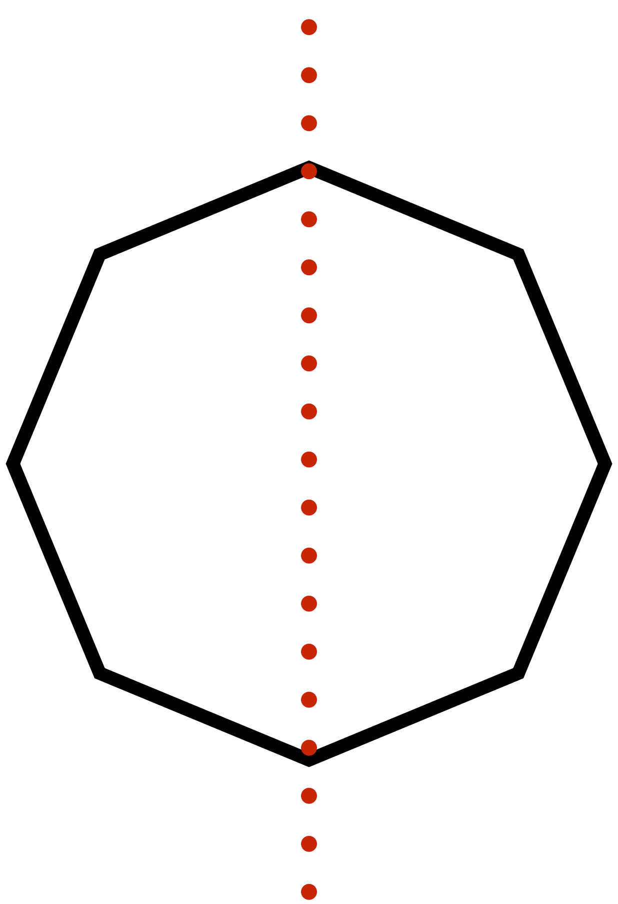
$$G = \{(\pi_1, \pi_2, b) : \pi_1, \pi_2 \in S_n, b \in \{0,1\}\}$$
$$= S_n \wr S_2$$

Dihedral Group

Dihedral group D_N on $2N$ elements is

$$D_N = \langle r, s : r^2 = s^N = 1, rsr = s^{-1} \rangle$$

It is the group of symmetries of a regular N-gon.



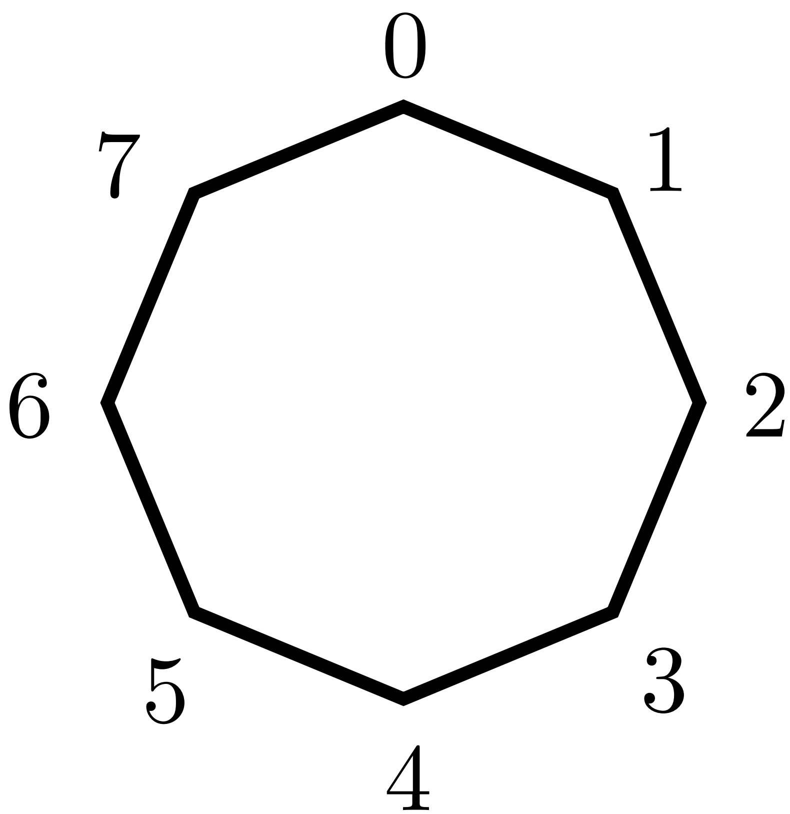
r is reflection about the dashed line.

Dihedral Group

Dihedral group D_N on $2N$ elements is

$$D_N = \langle r, s : r^2 = s^N = 1, rsr = s^{-1} \rangle$$

It is the group of symmetries of a regular N-gon.



s is a rotation by $\dfrac{2\pi}{N}$.

$$s(i) = i + 1 \bmod N$$

Dihedral Group

The dihedral group HSP is important because of its connection to the security of lattice based cryptography.

Kuperberg has given a quantum algorithm for the dihedral HSP that runs in time $2^{O(\sqrt{\log(N)})}$.

Bottlenecks

What are the obstacles to solving the non-abelian HSP?

We know efficient quantum circuits for the FT over many but not all non-abelian groups.

In particular, we know this for the symmetric and dihedral groups.

We can also use the same algorithm to produce a coset state.

The main bottleneck is efficiently extracting information from this coset state.

Query complexity of HSP

From an information theoretic perspective, there is enough info in coset states to identify H even in non-abelian groups.

Ettinger, Hoyer, and Knill [EHK99] show that for any finite group G there is a quantum algorithm for HSP that only makes $O(\log|G|)$ calls to the hiding function f.

The running time is still exponential in $\log|G|$.

EHK Query Algorithm

Let G be a finite group and $H \leq G$. Let $f : G \to [M]$ be a function that is constant on left cosets of H and has distinct values on distinct cosets.

As a test run, let us create the state

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |f(g)\rangle$$

and then measure and discard the second register. This leaves us in a coset state:

$$|aH\rangle = \frac{1}{\sqrt{|H|}} \sum_{h \in H} |ah\rangle$$

EHK Query Algorithm

For $L \leq G$ let $b_1, \ldots, b_t$ be such that $b_1 L, \ldots, b_t L$ form a partition of G and define

$$t = \frac{|G|}{|L|}$$

$$\Pi_L = \sum_{i=1}^{t} |b_i L\rangle\langle b_i L|$$

Claim 1: If $L \leq H$ then $\|\Pi_L |aH\rangle\|^2 = 1$.

If $L \not\leq H$ then $\|\Pi_L |aH\rangle\|^2 \leq \frac{1}{2}$.

Idea: For every $g \in G$ test if $g \in H$ by measuring with $\{\Pi_{\langle g\rangle}, \mathbb{I} - \Pi_{\langle g\rangle}\}$.

Amplification

To make this work we amplify the success probability: create the state

$$\frac{1}{\sqrt{|G|^m}} \sum_{g_1,\ldots,g_m \in G} |g_1,\ldots,g_m\rangle |f(g_1),\ldots,f(g_m)\rangle$$

This takes m queries.

Now measuring and discarding the second register we are in some tensor product of coset states:

$$|\psi\rangle = |a_1 H\rangle \cdots |a_m H\rangle$$

Amplification

$$|\psi\rangle = |a_1 H\rangle \cdots |a_m H\rangle$$

By Claim 1, if $L \leq H$ then $\| \Pi_L^{\otimes m} |\psi\rangle \|^2 = 1$.

If $L \not\leq H$ then $\| \Pi_L^{\otimes m} |\psi\rangle \|^2 \leq \dfrac{1}{2^m}$.

Let $g_1, \ldots, g_{|G|}$ be an enumeration of the elements of G.

We successively apply the measurements $\{\Pi_{\langle g_i \rangle}^{\otimes m}, \mathbb{I} - \Pi_{\langle g_i \rangle}^{\otimes m}\}$ on $|\psi\rangle$ for $i = 1, \ldots, |G|$.

Success Probability

We successively apply the measurements $\{\Pi_{\langle g_i \rangle}^{\otimes m}, (\Pi_{\langle g_i \rangle}^{\otimes m})^{\perp}\}$ on $|\psi\rangle$ for $i = 1, \ldots, |G|$.

Define $P_i = \Pi_{\langle g_i \rangle}^{\otimes m}$ if $g_i \in H$ and $P_i = \mathbb{I} - \Pi_{\langle g_i \rangle}^{\otimes m}$ o.w.

Thus P_i always corresponds to the "good" outcome.

If $g_i \in H$ we see outcome of P_i with probability 1 .

If $g_i \notin H$ we see outcome of P_i with probability $1 - 1/2^m$.

Quantum Union Bound

If $g_i \in H$ we see outcome of P_i with probability 1 .

If $g_i \notin H$ we see outcome of P_i with probability $1 - 1/2^m$.

Quantum Union Bound [Gao14]: We see the outcomes corresponding to $P_1, \ldots, P_{|G|}$ with probability at least

$$1 - \frac{4|G|}{2^m}$$

Taking $m = 2(\log(|G|) + 1)$ we succeed with probability at least $1 - 1/|G|$.

Proof of Claim

Claim

Now we prove claim 1.

Let $b_1 L, \ldots, b_t L$ form a partition of G and define

$$\Pi_L = \sum_{i=1}^{t} |b_i L\rangle\langle b_i L|$$

Claim 1: If $L \leq H$ then $\| \Pi_L |aH\rangle \|^2 = 1$.

If $L \not\leq H$ then $\| \Pi_L |aH\rangle \|^2 \leq \frac{1}{2}$.

Intersection

We first need a fact. If $L, H \leq G$ and $L \not\leq H$ then

$$|L \cap H| \leq |L|/2$$

This is because $L \cap H \leq L$ and so by Lagrange's theorem $|L \cap H|$ divides $|L|$.

As $L \cap H \neq L$ this means $|L \cap H| \leq |L|/2$.

Intersection

Let $X, Y \subseteq G$.

Define $|X\rangle = \dfrac{1}{\sqrt{|X|}} \displaystyle\sum_{x \in X} |x\rangle$ and similarly for $|Y\rangle$.

Then $\langle X|Y\rangle = \dfrac{|X \cap Y|}{\sqrt{|X||Y|}}$.

Coset Overlap

Let $K, L \leq G$ and $d = |K \cap L|$.

Then $|aK \cap bL| \in \{0, d\}$ (Q5 on PS2).

Now let $b_1, \ldots, b_t$ be such that $b_1 L, b_2 L, \ldots, b_t L$ form a partition of G (so $t = |G|/|L|$).

For how many i does $|aK \cap b_i L| = d$?

Coset Overlap

Let $K, L \leq G$ and let $aK = \{a \cdot k : k \in K\}$ be a left coset of K, and similarly bL a left coset of L.

For how many i does $|aK \cap b_i L| = d$?

It must be $|K|/d$ because every element of aK appears in some $b_i L$.

Proof of Claim

Define the projector $\Pi_L = \sum_{i=1}^{t} |b_i L\rangle\langle b_i L|$.

$\mathcal{I} = \{i : b_i L \cap aH \neq \emptyset\}$

$|\mathcal{I}| = \frac{|H|}{|L \cap H|}$

What is $\| \Pi_L |aH\rangle \|^2$?

$$\Pi_L |aH\rangle = \sum_{i=1}^{t} |b_i L\rangle\langle b_i L | aH\rangle$$

$$= \sum_{i \in \mathcal{I}} \frac{|L \cap H|}{\sqrt{|L||H|}} |b_i L\rangle$$

So $\| \Pi_L |aH\rangle \|^2 = \frac{|H|}{|L \cap H|} \frac{|L \cap H|^2}{|L||H|} = \frac{|L \cap H|}{|L|}$

Proof of Claim

Define the projector $\Pi_L = \sum_{i=1}^{t} |b_i L\rangle\langle b_i L|$.

$$\|\Pi_L |aH\rangle\|^2 = \frac{|L \cap H|}{|L|} = \begin{cases} 1 & L \leq H \\ \leq 1/2 & L \not\leq H \end{cases}$$

Abelian HSP
(bonus slides)

Abelian HSP

Let's treat the general abelian case now.

We are given a group G, a subgroup H and a function f with the promise that $f(x) = f(y)$ iff $x - y \in H$.

We assume that we can apply F_G the Fourier transform over G.

For example, this would be the case if we know the decomposition

$$G \cong \mathbb{Z}_{k_1} \times \cdots \times \mathbb{Z}_{k_t}$$

Algorithm

Step 1: Create uniform superposition over group elts.

$$|0\rangle|0\rangle \mapsto \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle|0\rangle$$

Step 2: Apply f .

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle|0\rangle \mapsto \frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle|f(g)\rangle$$

Algorithm

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |f(g)\rangle$$

Step 3: Measure and discard the second register. Then we have a coset state.

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

Apply the Fourier transform over G .

Fourier Transform

Let $\hat{G}$ denote the set of characters of G .

Recall a character $\chi : G \to \mathbb{C}$ satisfies

$$\chi(x+y) = \chi(x)\chi(y) \quad \text{for all} \quad x, y \in G$$

(writing the group additively).

$$F_G|x\rangle = \frac{1}{\sqrt{|G|}} \sum_{\chi \in \hat{G}} \chi(x)^* |\chi\rangle$$

Apply Fourier Transform

Let $\hat{G}$ denote the set of characters of G .

$$F_G \frac{1}{\sqrt{|H|}} \sum_{h \in H} |x+h\rangle = \frac{1}{\sqrt{|G||H|}} \sum_{h \in H} \sum_{\chi \in \hat{G}} \chi(x+h)^* |\chi\rangle$$

$$= \frac{1}{\sqrt{|G||H|}} \sum_{\chi \in \hat{G}} \chi(x)^* |\chi\rangle \sum_{h \in H} \chi(h)^*$$

$$\frac{1}{\sqrt{|G||H|}} \sum_{\chi \in \hat{G}} \chi(x)^* |\chi\rangle \sum_{h \in H} \chi(h)^*$$

Let $H^\perp \subseteq \hat{G}$ be the set of characters χ with $\chi(h) = 1$ for all $h \in H$.

Key property:

$$\sum_{h \in H} \chi(h)^* = \begin{cases} |H| & \text{if } \chi \in H^\perp \\ 0 & \text{otherwise} \end{cases} .$$

Classical Post-Processing

Applying the key property the final state is:

$$\frac{1}{\sqrt{|G||H|}} \sum_{\chi \in \hat{G}} \chi(x)^* |\chi\rangle \sum_{h \in H} \chi(h)^* = \frac{\sqrt{|H|}}{\sqrt{|G|}} \sum_{\chi \in H^\perp} \chi(x)^* |\chi\rangle$$

Note the normalization tells us $|H^\perp| = |G|/|H|$.

Now we measure and get the name of a character $\chi \in H^\perp$.

Let $\ker(\chi) = \{g \in G : \chi(g) = 1\}$. We gain the info

$$H \subseteq \ker(\chi)$$

Classical Post-Processing

We now repeat this process several times and say we see $\chi_1, \ldots, \chi_t$.

We know that $H \subseteq \ker(\chi_1) \cap \cdots \cap \ker(\chi_t)$.

How many times do we have to repeat until with good probability this becomes an equality?

Let $K = \ker(\chi_1) \cap \cdots \cap \ker(\chi_t)$. This is a subgroup of G containing H .

Classical Post-Processing

Let $K = \ker(\chi_1) \cap \cdots \cap \ker(\chi_t)$. This is a subgroup of G containing H .

Suppose $H < K$. By Lagrange's theorem $|H|$ divides $|K|$ and so $|K| > 2|H|$.

For a subgroup L we have $|L^\perp| = |G|/|L|$.

Thus with prob. at least $1/2$ a random $\chi \in H^\perp$ is not in $K^\perp$.

Then $|K \cap \ker(\chi)| \leq |K|/2$.

Classical Post-Processing

Thus so long as $H < K$ each repetition reduces the size of K by at least $1/2$.

Thus whp after $O(\log(|G|))$ many repetitions we identify H.