[Code Equivalence is Hard for Shor-like Quantum Algorithms]

Hang Dinh – Indiana University South Bend

joint work with

Cristopher Moore – University of New Mexico

Alexander Russell – University of Connecticut

Webinar on Symbolic Computations and Post-Quantum Cryptography
(SCPQ Webinar May 17, 2012)

Outline

- Overview/Motivation
  - Code Equivalence
  - Why care?
- Shor-like algorithms
  - Quantum Fourier Sampling (QFS)
  - Hidden Subgroup Problems (HSP)
- Reduction from Code Equivalence to HSP
- Our results
  - General results
  - Codes that make Code Equivalence hard for QFS

Code Equivalence (CE)

- **Code Equivalence** [Petrank and Roth, 1997]
  - Given the generator matrices of two linear codes C and C’
  - Decide if C is equivalent to C’ up to a permutation of the codeword coordinates
- A search variant of CE:
  - Find a permutation between two given equivalent codes
- Hardness [Petrank and Roth, 1997]
  - Code Equivalence is unlikely NP-complete,
  - but at least as hard as Graph Isomorphism
    - There’s an efficient reduction from Graph Isomorphism to CE

CE and Code-based Cryptosystems

	McEliece systems	Neiderreiter systems
Secret code C	q-ary $[n, k]$-code	q-ary $[n, n - lk]$-code
Secret key	M: $k \times n$ generator matrix of C	M: $k \times n$ parity check matrix over $\mathbf{F}_{q^l}$ of C
	S: $k \times k$ invertible matrix over $\mathbf{F}_q$	
	P: $n \times n$ permutation matrix	
Public key	$M' = SMP$	

- If the secret code is known to the adversary
  - recover secret key S and P → solve CE for the secret code

CE and Code-based Cryptosystems

- The secret code can be known to the adversary
  - if the space of all codes of the same parameters (q, n, k) and same family as the secret code is small.
- Example: Reed-Muller codes (q=2)
  - used in the Sildelnikov cryptosystem [Sidelnikov, 1994]
  - there's a *single* Reed-Muller code of given length and dimension.
- Example: *special* binary Goppa codes
  - those generated by polynomials of binary coefficients
  - can exhaustively search [Loidreau and Sendrier, 2001]

Best Known Algorithm for CE

- Support Splitting Algorithm [Sendrier, 1999]
  - Classical, deterministic
  - Efficient for **binary** codes with small hull dimension, including binary Goppa codes.
  - *Likely* to be efficient for non-binary codes with small hull dimension
  - **Inefficient** for other codes, such as Reed-Muller codes.

Can Quantum Algorithms Do Better?

- The most popular paradigm of quantum algorithms
  - generalize Shor’s algorithms
  - reply on quantum Fourier transform
  - solve the class of hidden subgroup problems (HSP).
  - Nearly all known quantum algorithms that provide exponential speedup are designed in this paradigm.

- There’s a natural reduction from CE to HSP
  - So, can CE be solved efficiently by Shor-like algorithms?

Outline

- Overview/Motivation
  - Code Equivalence
  - Why care?
- **Shor-like algorithms**
  - Quantum Fourier Sampling (QFS)
  - Hidden Subgroup Problems (HSP)
- Reduction from Code Equivalence to HSP
- Our results
  - General results
  - Codes that make Code Equivalence hard for QFS

Hidden Subgroup Problem (HSP)

- HSP over a finite group G:
  - Input: a black-box function f on G that *separates* the left (or right) cosets of an unknown subgroup $H < G$, i.e.,
    $$f(x) = f(y) \text{ iff } xH = yH$$
  - Output: a generating set for H.
- Well-known interesting cases
  - HSP over cyclic groups $\mathbf{Z}_N$ → factorization
  - HSP over $\mathbf{Z}_N \times \mathbf{Z}_N$ → discrete logarithm
  - HSP over symmetric groups S_n → Graph Isomorphism
  - HSP over dihedral groups D_n → unique-Shortest-vector

Shor-like Algorithms

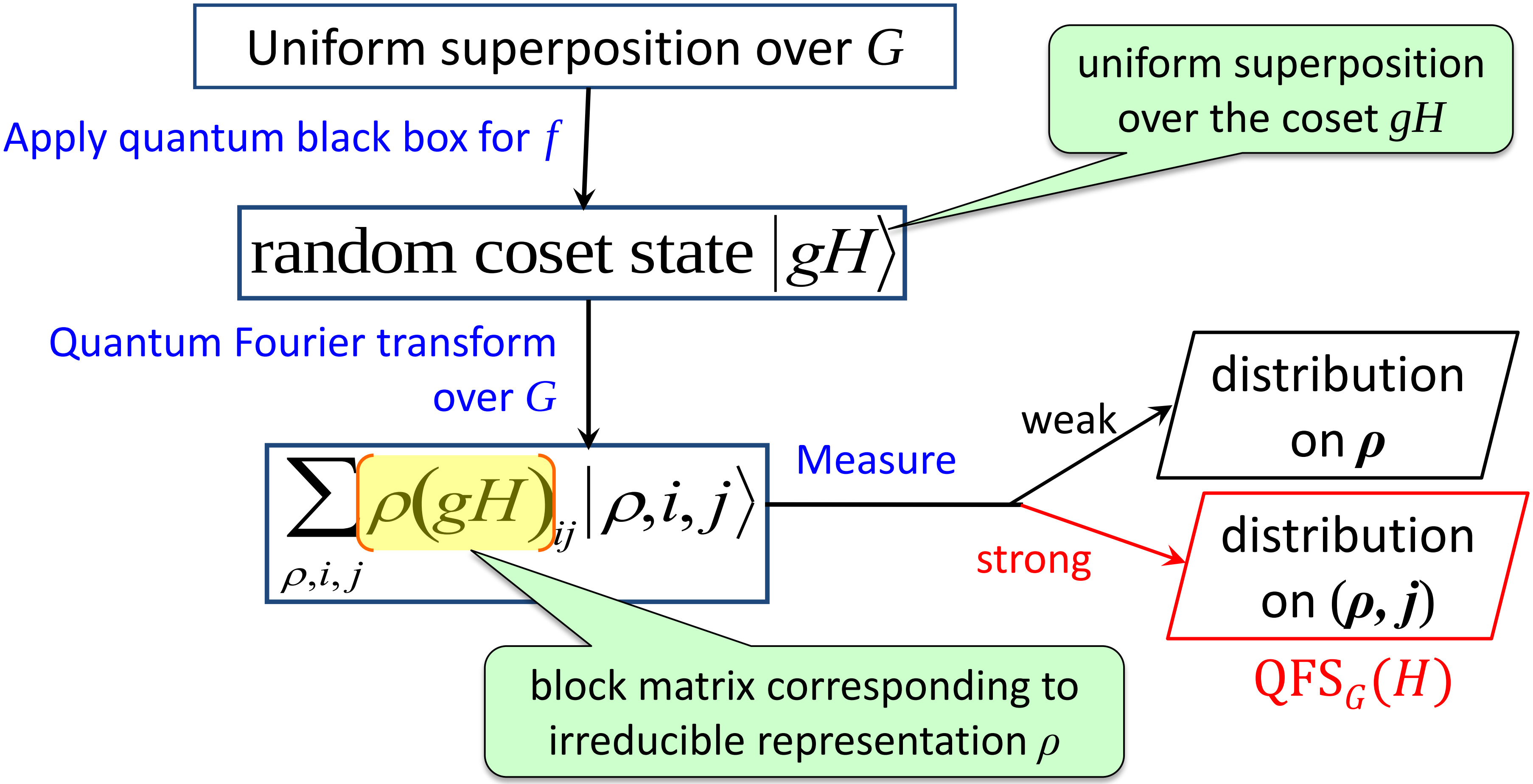
- To solve the HSP over G with hidden subgroup H

Quantum Fourier Sampling (QFS) over G using back box f that separates cosets of H

a probability distribution, denoted $\mathrm{QFS}_G(H)$

Classically recover H using information from the distribution $\mathrm{QFS}_G(H)$

Quantum Fourier Sampling (QFS)



Efficiency of Shor-like Algorithms

- QFS is efficient for HSP over abelian groups.
- Some nonabelian HSPs *may* be efficiently solvable
  - They have efficient quantum Fourier transforms.
  - Subexponential time for dihedral HSP [Kuperberg, 2003]
- Strong QFS doesn't work for S_n if $|H| = 2$
  - it can't distinguish among conjugates of H and the trivial one
  - i.e., $\mathrm{QFS}_G(gHg^{-1})$ is close to $\mathrm{QFS}_G(\{1\})$, for most $g \in G$.
  - [Moore, Russell, Schulman, 2008].

Outline

- Overview/Motivation
  - Code Equivalence
  - Why care?
- Shor-like algorithms
  - Quantum Fourier Sampling (QFS)
  - Hidden Subgroup Problems (HSP)
- **Reduction from Code Equivalence to HSP**
- Our results
  - General results
  - Codes that make Code Equivalence hard for QFS

Reduce CE to HSP

Search variant of Code Equivalence

Scrambler-Permutation Problem

Hidden Shift Problem

Hidden Subgroup Problem

CE to Scrambler-Permutation

- **Scrambler-Permutation Problem**
  - Input: $k \times n$ matrices M and M' over a field $\mathbf{F}_{q^l} \supseteq \mathbf{F}_q$ such that $M' = SMP$ for some $(S, P) \in \mathrm{GL}_k(\mathbf{F}_q) \times S_n$
  - Output: (S, P)
- Special case: attacking McEliece systems
  - $l = 1 \quad (\mathbf{F}_{q^l} = \mathbf{F}_q)$
  - M is a generator matrix of a q-ary $[n, k]$-code.
- Special case: attacking Neiderreiter systems
  - M is parity check matrix of a q-ary $[n, n - lk]$-code.

Scrambler-Permutation to Hidden Shift

- **Hidden Shift Problem** over a finite group G:
  - Input: two functions f_1, f_2 on G s.t. $\exists s \in G$ satisfying
    $$f_1(sg) = f_2(g) \text{ for all } g \in G$$
  - Output: a hidden shift s

Input: M and $M' = SMP$. Output: $(S, P) \in \mathrm{GL}_k(\mathbf{F}_q) \times S_n$

$\downarrow$

Hidden Shift Problem over $\mathrm{GL}_k(\mathbf{F}_q) \times S_n$

- Input: $f_1(X, Y) = X^{-1}MY$ and $f_2(X, Y) = X^{-1}M'Y$
- Output: a hidden shift (S^{-1}, P)

Hidden Shift to Hidden Subgroup

Hidden Shift Problem over a finite group G:

- Input: two functions f_1, f_2 on G s.t. $\exists s \in G$ satisfying

$$f_1(sg) = f_2(g) \text{ for all } g \in G$$

- Output: a hidden shift s

HSP over wreath product $G \wr \mathbf{Z}_2$ (semidirect product of G^2 and $\mathbf{Z}_2$)

- Input: function f defined as:

$$f\big((g_1, g_2), 0\big) = \big(f_1(g_1), f_2(g_2)\big)$$
$$f\big((g_1, g_2), 1\big) = \big(f_2(g_2), f_1(g_1)\big)$$

Hidden Shift to Hidden Subgroup

Hidden Shift Problem over a finite group G:

- Input: two functions f_1, f_2 on G s.t. $\exists s \in G$ satisfying

$$f_1(sg) \;=\; f_2(g) \text{ for all } g \in G$$

- Output: a hidden shift s

HSP over wreath product $G \wr \mathbf{Z}_2$(semidirect product of G^2 and $\mathbf{Z}_2$)

- Output: subgroup $H = ((H_0, s^{-1}H_0 s), 0) \cup ((H_0 s, s^{-1}H_0), 1)$ where

$$H_0 = \{g \in G | f_1(g) = f_1(1)\} < G$$
$$H_0 s = \text{The set of all hidden shifts}$$

f_1 must separate right cosets of H_0

Scrambler-Permutation to HSP

Scrambler-Permutation Problem

- Input: M and $M' = SMP$ for some $(S, P) \in \mathrm{GL}_k(\mathbf{F}_q) \times S_n$
- Output: (S, P)

HSP over the wreath product $(\mathrm{GL}_k(\mathbf{F}_q) \times S_n) \wr \mathbf{Z}_2$

- hidden subgroup: $H = ((H_0, s^{-1} H_0 s), 0) \cup ((H_0 s, s^{-1} H_0), 1)$

  where

  $$H_0 = \{(S, P) | S^{-1} M P = M\} < \mathrm{GL}_k(\mathbf{F}_q) \times S_n$$

  $$s = (S^{-1}, P)$$

Can this HSP be solved efficiently by strong QFS?
Can QFS distinguish conjugates gHg^{-1} and $\{1\}$?

Outline

- Overview/Motivation
  - Code Equivalence
  - Why care?
- Shor-like algorithms
  - Quantum Fourier Sampling (QFS)
  - Hidden Subgroup Problems (HSP)
- Reduction from Code Equivalence to HSP
- **Our results**
  - General results
  - Codes that make Code Equivalence hard for QFS

Our Results

- We show that in many cases of interest,
  - $\mathrm{QFS}_G(gHg^{-1})$ is exponentially close to $\mathrm{QFS}_G(\{1\})$, for most $g \in G$.
  - In such a case, H is called *indistinguishable* by strong QFS.
- Apply to $G = S_n$ with $|H| \geq 2$
- Apply to the CE for many codes, including
  - Goppa codes, generalized Reed-Solomon codes
    [Dinh, Moore, Russell, CRYPTO 2011]
  - Reed-Muller codes
    [Dinh, Moore, Russell, Preprint 2011 , arXiv:1111.4382]

Hidden Symmetries

- Recall: the hidden subgroup reduced from matrix M is determined by the subgroup
$$H_0 = \{(S,P) | S^{-1}MP = M\} < \mathrm{GL}_k(\mathbf{F}_q) \times S_n$$
- Projection of H_0 onto S_n is the *automorphism group*
$$\mathrm{Aut}(M) \coloneqq \{P \in S_n | \exists S \in \mathrm{GL}_k(\mathbf{F}_q), SMP = M\}$$
  - Each $P \in \mathrm{Aut}(M)$ has the same number N of preimages $S \in \mathrm{GL}_k(\mathbf{F}_q)$ in this projection.
  - <u>Fact</u>: Let r be the column rank of M. Then $N \leq q^{lk(k-r)}$.
  - Hence, $|H_0| \leq |\mathrm{Aut}(M)|\, q^{lk(k-r)}$.

General Results for CE

- Theorem [Dinh, Moore, Russell, CRYPTO 2011]:
  - Assume $k^2 \leq 0.2n \log_q n$.
  - The hidden subgroup reduced from matrix M is indistinguishable by strong QFS if

1) $|\mathrm{Aut}(M)| \leq e^{o(n)}$
2) The *minimal degree* of $\mathrm{Aut}(M)$ is $\geq \Omega(n)$.
3) The column rank of of M is $\geq k - o(\sqrt{n})/l$.

The *minimal degree* of $\mathrm{Aut}(M)$ is the minimal number of points *moved* by a non-identity permutation in $\mathrm{Aut}(M)$.

HSP-hard Codes

- What codes make CE hard for Shor-like algorithms?
  - A linear code is called *HSP-hard* if it has a generator matrix or parity check matrix M s.t. the hidden subgroup reduced from M is indistinguishable by strong QFS.
- Observe: If M is a generator matrix of a code C
  - Then $\mathrm{Aut}(M) = \mathrm{Aut}(C)$, and M has full rank.
- Corollary: Let C be a q-ary $[n, k]$-code such that $k^2 \leq 0.2n \log_q n$. Then C is HSP-hard if
  1) $|\mathrm{Aut}(C)| \leq e^{o(n)}$
  2) The minimal degree of $\mathrm{Aut}(C)$ is $\geq \Omega(n)$.

Reed-Muller Codes are HSP-hard

- Reed-Muller code $\mathrm{RM}(r,m)$

  $= \{(f(\alpha_1),\dots,f(\alpha_n)) \mid f \in \mathbf{F}_2[X_1,..,X_m], \deg(f) \le r\}$,

  where $(\alpha_1,\dots,\alpha_n)$ is a fixed ordered list of all vectors in ${\mathbf{F}_2}^m$

  – has length $n = 2^m$ and dimension $k = \sum_{j=0}^{r} \binom{m}{j}$.

  – If $r < 0.1m$, then $k < r\binom{m}{0.1m} < r2^{0.47m}$, and $k^2 \le 0.2nm$ for sufficiently large m.

- <u>Theorem</u>: Reed-Muller codes $\mathrm{RM}(r,m)$ with $r < 0.1m$ and m sufficiently large are HSP-hard.

Automorphism Group of Reed-Muller Codes

- Fact:

  $\mathrm{Aut}\big(\mathrm{RM}(r,m)\big) =$ general affine group of space $\mathbf{F}_2{}^m$

  $= \left\{\sigma_{A,\boldsymbol{b}}: \mathbf{F}_2{}^m \to \mathbf{F}_2{}^m, \sigma_{A,\boldsymbol{b}}(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{b} \,\middle|\, A \in \mathrm{GL}_m(\mathbf{F}_2), \boldsymbol{b} \in \mathbf{F}_2{}^m\right\}$

- Propositions:

  1. $\left|\mathrm{Aut}\big(\mathrm{RM}(r,m)\big)\right| = |\mathrm{GL}_m(\mathbf{F}_2)| \times |\mathbf{F}_2{}^m| \leq 2^{m^2+m}$

     $\leq 2^{O\left(\log^2 n\right)} \leq e^{o(n)}$, where $n = 2^m$

  2. The minimal degree of $\mathrm{Aut}(\mathrm{RM}(r,m))$ is exactly 2^{m-1}.

Automorphism Group of Reed-Muller Codes

2a. The minimal degree of $\mathrm{Aut}(\mathrm{RM}(r,m))$ is $\leq 2^{m-1}$.

Recall: min deg. of $\mathrm{Aut}(C)$:= $\min\{\mathrm{supp}(\pi) | \pi \in \mathrm{Aut}(C), \pi \neq \mathrm{Id}\}$,

where $\mathrm{supp}(\pi) \coloneqq \#\{i: \pi(i) \neq i\}$.

Proof:

– An affine transformation $\sigma_{A,\mathbf{0}}: \mathbf{F}_2{}^m \to \mathbf{F}_2{}^m$ with support 2^{m-1}

$$\sigma_{A,\mathbf{0}}(\boldsymbol{x}) = A\boldsymbol{x} = \begin{pmatrix} \mathbf{1} & & & \mathbf{1} \\ & \mathbf{1} & \ddots & \\ & & \mathbf{1} & \\ & & & \mathbf{1} \end{pmatrix} \boldsymbol{x}$$

– This $\sigma_{A,\mathbf{0}}$ fixes all vectors $\boldsymbol{x} \in \mathbf{F}_2{}^m$ with $x_m = 0$.

– There are $2^m - 2^{m-1} = 2^{m-1}$ vectors not fixed by $\sigma_{A,\mathbf{0}}$

Automorphism Group of Reed-Muller Codes

2b. The minimal degree of $Aut(\mathrm{RM}(r, m))$ is $\geq 2^{m-1}$.

- Claim 1: If $\sigma_{A,\boldsymbol{b}}$ fixes a set S that spans $\mathbf{F}_2{}^m$, then $\sigma_{A,\boldsymbol{b}} = \mathrm{Id}$.

- Claim 2: Any set $S \subseteq \mathbf{F}_2{}^m$ with size $> 2^{m-1}$ spans $\mathbf{F}_2{}^m$.

→No none-identity affine transformation can fix >2^{m-1} vectors.

Automorphism Group of Reed-Muller Codes

2b. The minimal degree of $Aut(\mathrm{RM}(r,m))$ is $\geq 2^{m-1}$.

– <u>Claim 1</u>: If $\sigma_{A,\boldsymbol{b}}$ fixes a set S that spans $\mathbf{F}_2{}^m$, then $\sigma_{A,\boldsymbol{b}} = \mathrm{Id}$.

Proof: Let $\boldsymbol{s} \in S$ and $S' = S - \boldsymbol{s}$. Then S' also spans $\mathbf{F}_2{}^m$, and A fixes S', in which case $A = \mathbf{1}$. Then $\boldsymbol{b} = \mathbf{0}$. Note $\sigma_{\mathbf{1},\mathbf{0}} = \mathrm{Id}$.

– <u>Claim 2</u>: Any set $S \subseteq \mathbf{F}_2{}^m$ with size $> 2^{m-1}$ spans $\mathbf{F}_2{}^m$.

→No none-identity affine transformation can fix >2^{m-1} vectors.

Automorphism Group of Reed-Muller Codes

2b. The minimal degree of $Aut(\mathrm{RM}(r, m))$ is $\geq 2^{m-1}$.

– <u>Claim 1</u>: If $\sigma_{A,\boldsymbol{b}}$ fixes a set S that spans $\mathbf{F}_2{}^m$, then $\sigma_{A,\boldsymbol{b}} = \mathrm{Id}$.

Proof: Let $\boldsymbol{s} \in S$ and $S' = S - \boldsymbol{s}$. Then S' also spans $\mathbf{F}_2{}^m$, and A fixes S', in which case $A = \mathbf{1}$. Then $\boldsymbol{b} = \mathbf{0}$. Note $\sigma_{\mathbf{1},\mathbf{0}} = \mathrm{Id}$.

– <u>Claim 2</u>: Any set $S \subseteq \mathbf{F}_2{}^m$ with size $> 2^{m-1}$ spans $\mathbf{F}_2{}^m$.

Proof: Let $B \subseteq S$ be a maximal set that consists of linearly independent vectors. Since B spans S, $2^{|B|} \geq |S| > 2^{m-1}$. Then $|B| = m$. So B, and therefore S, spans $\mathbf{F}_2{}^m$.

→No none-identity affine transformation can fix >2^{m-1} vectors.

Open Question and Notes

- Are there other HSP-hard codes that are of cryptographic interest?
- Cautionary notes
  - Shor-like algorithms are unlikely to help break code-based cryptosystems using HSP-hard codes.
  - **But** we have not shown that other quantum algorithms, or even classical ones, cannot break code-based cryptosystems.
  - Nor have we shown that such an algorithm would violate a natural hardness assumption (such as lattice-based cryptosystems and Learning With Errors).