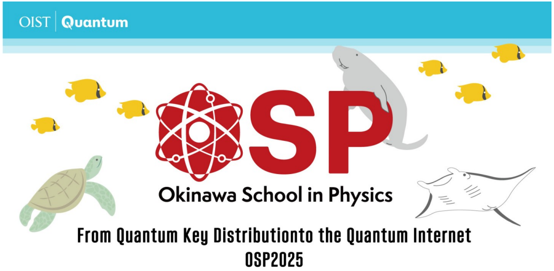
Quantum Cryptanalysis

Manuel Goulão

manuel.goulao@inesc-id.pt



Overview

Contents

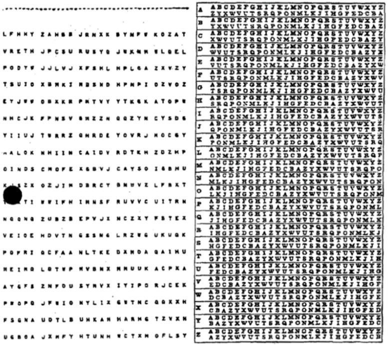
Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers



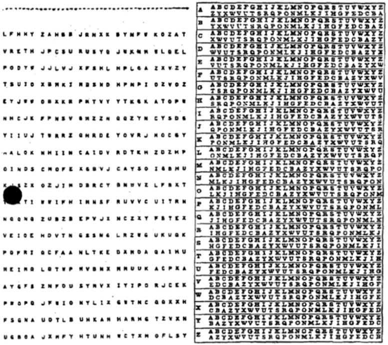
NSA One-Time Pad (Source: Wikimedia)

Focus on *computationally secure* protocols

Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)



NSA One-Time Pad (Source: Wikimedia)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers

Focus on *computationally secure* protocols

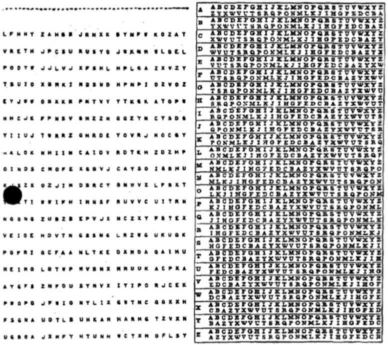
Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers



NSA One-Time Pad (Source: Wikimedia)

Focus on *computationally secure* protocols

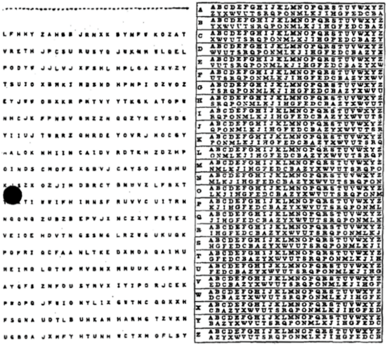
Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers



NSA One-Time Pad (Source: Wikimedia)

Focus on *computationally secure* protocols

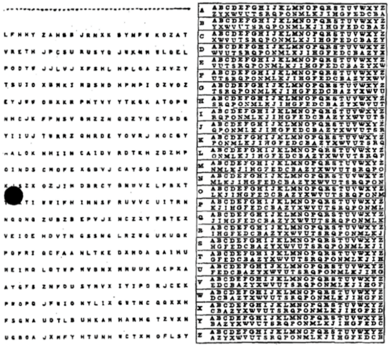
Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers



NSA One-Time Pad (Source: Wikimedia)

Focus on *computationally secure* protocols

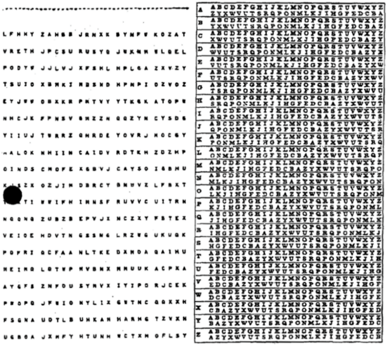
Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers



NSA One-Time Pad (Source: Wikimedia)

Focus on *computationally secure* protocols

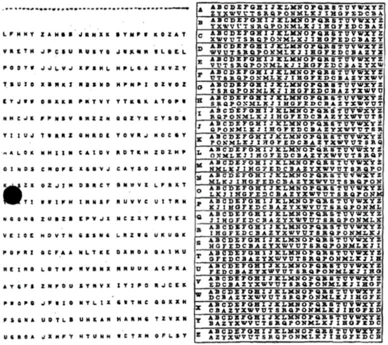
Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers



NSA One-Time Pad (Source: Wikimedia)

Focus on *computationally secure* protocols

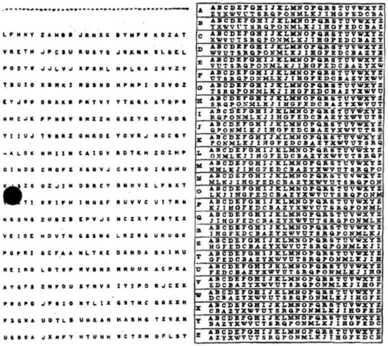
Cryptography & Cryptanalysis

How to communicate securely?

- Unconditional security (e.g., OTP)
- Computational security (e.g., RSA)

Cryptanalysis

- By hand
- Early automata
- Classical computers
- Quantum computers



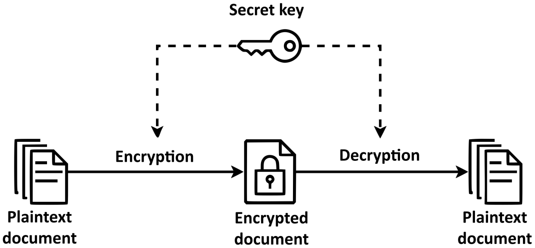
NSA One-Time Pad (Source: Wikimedia)

Focus on *computationally secure* protocols

Computational Cryptography

Symmetric Cryptosystems

- 1 shared secret key
- AES, SHA, etc.
- *Computational assumptions:* highly unstructured/nonlinear problems

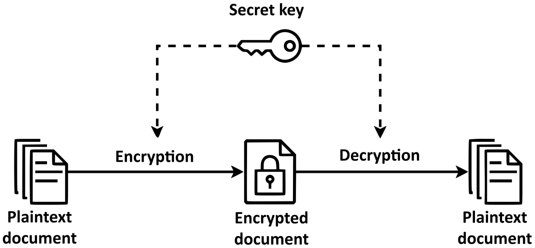


Symmetric Encryption (Source: Wikipedia)

Computational Cryptography

Symmetric Cryptosystems

- 1 shared secret key
- AES, SHA, etc.
- *Computational assumptions:* highly unstructured/nonlinear problems

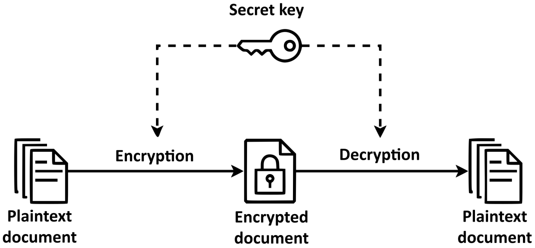


Symmetric Encryption (Source: Wikipedia)

Computational Cryptography

Symmetric Cryptosystems

- 1 shared secret key
- AES, SHA, etc.
- *Computational assumptions:* highly unstructured/nonlinear problems

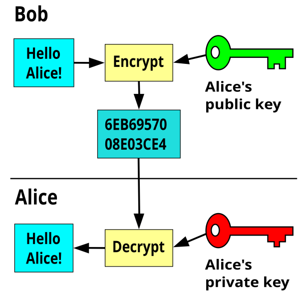


Symmetric Encryption (Source: Wikipedia)

Computational Cryptography

Asymmetric Cryptosystems

- 1 secret key & 1 public key
- RSA, DSA, DH, etc.
- *Computational assumptions:* algebraic problems with lots of structure

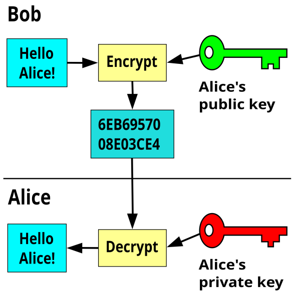


Assymetric Encryption (Source: Wikipedia)

Computational Cryptography

Asymmetric Cryptosystems

- 1 secret key & 1 public key
- RSA, DSA, DH, etc.
- *Computational assumptions:* algebraic problems with lots of structure

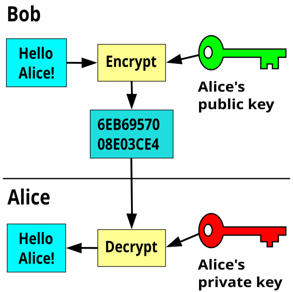


Assymetric Encryption (Source: Wikipedia)

Computational Cryptography

Asymmetric Cryptosystems

- 1 secret key & 1 public key
- RSA, DSA, DH, etc.
- *Computational assumptions:* algebraic problems with lots of structure



Assymetric Encryption (Source: Wikipedia)

Computational Cryptography

Current Deployment

- PKC is ubiquitous in the information-world (internet, credit cards, messaging, etc.)
- *Harvest/Store Now Decrypt Later* (HNDL/SNDL)
- Push to standardize post-quantum — already in TLS (hybrid)

RSA (factoring)

PK: $1 < e < \phi(p \cdot q)$;
SK: $d \equiv e^{-1} \pmod{\phi(p \cdot q)}$

Encryption: $c \equiv m^e \pmod{p \cdot q}$

Decryption: $m \equiv c^d \pmod{p \cdot q}$

El Gamal (dlog)

SK: $x \in \mathbb{Z}_q$, **PK**: $h = g^x$

Encryption: $c_1 = g^k$, $c_2 = m \cdot h^k$

Decryption: $m = c_2 \cdot c_1^{-x}$

Computational Cryptography

Current Deployment

- PKC is ubiquitous in the information-world (internet, credit cards, messaging, etc.)
- *Harvest/Store Now Decrypt Later* (HNDL/SNDL)
- Push to standardize post-quantum — already in TLS (hybrid)

RSA (factoring)

PK: $1 < e < \phi(p \cdot q)$;
SK: $d \equiv e^{-1} \pmod{\phi(p \cdot q)}$

Encryption: $c \equiv m^e \pmod{p \cdot q}$

Decryption: $m \equiv c^d \pmod{p \cdot q}$

El Gamal (dlog)

SK: $x \in \mathbb{Z}_q$, **PK**: $h = g^x$

Encryption: $c_1 = g^k$, $c_2 = m \cdot h^k$

Decryption: $m = c_2 \cdot c_1^{-x}$

Computational Cryptography

Current Deployment

- PKC is ubiquitous in the information-world (internet, credit cards, messaging, etc.)
- *Harvest/Store Now Decrypt Later* (HNDL/SNDL)
- Push to standardize post-quantum — already in TLS (hybrid)

RSA (factoring)

PK: $1 < e < \phi(p \cdot q)$;
SK: $d \equiv e^{-1} \pmod{\phi(p \cdot q)}$

Encryption: $c \equiv m^e \pmod{p \cdot q}$

Decryption: $m \equiv c^d \pmod{p \cdot q}$

El Gamal (dlog)

SK: $x \in \mathbb{Z}_q$, **PK**: $h = g^x$

Encryption: $c_1 = g^k$, $c_2 = m \cdot h^k$

Decryption: $m = c_2 \cdot c_1^{-x}$

Computational Cryptography

Current Deployment

- PKC is ubiquitous in the information-world (internet, credit cards, messaging, etc.)
- *Harvest/Store Now Decrypt Later* (HNDL/SNDL)
- Push to standardize post-quantum — already in TLS (hybrid)

RSA (factoring)

PK: $1 < e < \phi(p \cdot q)$;
SK: $d \equiv e^{-1} \pmod{\phi(p \cdot q)}$

Encryption: $c \equiv m^e \pmod{p \cdot q}$

Decryption: $m \equiv c^d \pmod{p \cdot q}$

El Gamal (dlog)

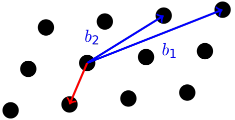
SK: $x \in \mathbb{Z}_q$, **PK**: $h = g^x$

Encryption: $c_1 = g^k$, $c_2 = m \cdot h^k$

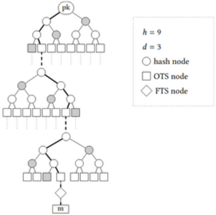
Decryption: $m = c_2 \cdot c_1^{-x}$

Security of post-quantum schemes

- **NIST Standard** (2022-25):
  - Kyber (Lattice, KEM)
  - HQC (Code, KEM)
  - Dilithium, FALCON (Lattice, DS)
  - SPHINCS+ (Hash, DS)



Source: Wikipedia



Source: SPHINCS+

- Not thoroughly studied classically...
- ... even less for **quantum attacks**.

Source: Sendrier SP'17

Security of post-quantum schemes

- **NIST Standard** (2022-25):
  - Kyber (Lattice, KEM)
  - HQC (Code, KEM)
  - Dilithium, FALCON (Lattice, DS)
  - SPHINCS+ (Hash, DS)

- Not thoroughly studied classically...
- ... even less for **quantum attacks**.

Breaking Rainbow Takes a Weekend on a Laptop

Ward Beullens

IBM Research, Zurich, Switzerland
wbe@zurich.ibm.com

Abstract. This work introduces new key recovery attacks against the Rainbow signature scheme, which is one of the three finalist signature schemes still in the NIST Post-Quantum Cryptography standardization project. The new attacks outperform previously known attacks for all the parameter sets submitted to NIST and make a key-recovery practical for the SL 1 parameters. Concretely, given a Rainbow public key for the SL 1 parameters of the second-round submission, our attack returns the corresponding secret key after on average 53 hours (one weekend) of computation time on a standard laptop.

An efficient key recovery attack on SIDH

Wouter Castryck[1,2] and Thomas Decru[1]

[1] imec-COSIC, KU Leuven, Belgium
[2] Vakgroep Wiskunde: Algebra en Meetkunde, Universiteit Gent, Belgium

Abstract. We present an efficient key recovery attack on the Supersingular Isogeny Diffie–Hellman protocol (SIDH). The attack is based on Kani's "reducibility criterion" for isogenies from products of elliptic curves and strongly relies on the torsion point images that Alice and Bob exchange during the protocol. If we assume knowledge of the endomorphism ring of the starting curve then the classical running time is polynomial in the input size (heuristically), apart from the factorization of a small number of integers that only depend on the system parameters. The attack is particularly fast and easy to implement if one of the parties uses 2-isogenies and the starting curve comes equipped with a non-scalar endomorphism of very small degree; this is the case for SIKE, the instantiation of SIDH that recently advanced to the fourth round of NIST's standardization effort for post-quantum cryptography. Our Magma implementation breaks `SIKEp434`, which aims at security level 1, in about ten minutes on a single core.

Broken NIST-PQC finalists

Security of post-quantum schemes

- **NIST Standard** (2022-25):
  - Kyber (Lattice, KEM)
  - HQC (Code, KEM)
  - Dilithium, FALCON (Lattice, DS)
  - SPHINCS+ (Hash, DS)

- Not thoroughly studied classically...
- ... even less for **quantum attacks**.

Breaking Rainbow Takes a Weekend on a Laptop

Ward Beullens

IBM Research, Zurich, Switzerland
wbe@zurich.ibm.com

Abstract. This work introduces new key recovery attacks against the Rainbow signature scheme, which is one of the three finalist signature schemes still in the NIST Post-Quantum Cryptography standardization project. The new attacks outperform previously known attacks for all the parameter sets submitted to NIST and make a key-recovery practical for the SL 1 parameters. Concretely, given a Rainbow public key for the SL 1 parameters of the second-round submission, our attack returns the corresponding secret key after on average 53 hours (one weekend) of computation time on a standard laptop.

An efficient key recovery attack on SIDH

Wouter Castryck[1,2] and Thomas Decru[1]

[1] imec-COSIC, KU Leuven, Belgium
[2] Vakgroep Wiskunde: Algebra en Meetkunde, Universiteit Gent, Belgium

Abstract. We present an efficient key recovery attack on the Supersingular Isogeny Diffie–Hellman protocol (SIDH). The attack is based on Kani's "reducibility criterion" for isogenies from products of elliptic curves and strongly relies on the torsion point images that Alice and Bob exchange during the protocol. If we assume knowledge of the endomorphism ring of the starting curve then the classical running time is polynomial in the input size (heuristically), apart from the factorization of a small number of integers that only depend on the system parameters. The attack is particularly fast and easy to implement if one of the parties uses 2-isogenies and the starting curve comes equipped with a non-scalar endomorphism of very small degree; this is the case for SIKE, the instantiation of SIDH that recently advanced to the fourth round of NIST's standardization effort for post-quantum cryptography. Our Magma implementation breaks `SIKEp434`, which aims at security level 1, in about ten minutes on a single core.

Broken NIST-PQC finalists

Contents

Computational Cryptography

1. All parties are classical — computation is PPT:
   classical security
2. Adversary has a quantum computer — computation is QPT:
   post-quantum security
3. **Communication** is quantum (ptx/ctx/keys are still classical):
   quantum security (qCPA/qCCA/qCMA)

Computational Cryptography

1. All parties are classical — computation is PPT:

   classical security

2. Adversary has a quantum computer — computation is QPT:

   post-quantum security

3. **Communication** is quantum (ptx/ctx/keys are still classical):

   quantum security (qCPA/qCCA/qCMA)

Computational Cryptography

1. All parties are classical — computation is PPT:
   classical security
2. Adversary has a quantum computer — computation is QPT:
   post-quantum security
3. **Communication** is quantum (ptx/ctx/keys are still classical):
   quantum security (qCPA/qCCA/qCMA)

Computational Cryptography

1. All parties are classical — computation is PPT:
   classical security

2. Adversary has a quantum computer — computation is QPT:
   post-quantum security

3. **Communication** is quantum (ptx/ctx/keys are still classical):
   quantum security (qCPA/qCCA/qCMA)

Contents

Grover's Search

Grover's search

INPUT: $f : \{0, \dots, N-1\} \to \{0, 1\}$ where $f(x) = 1$ for a single x

OUTPUT: ω such that $f(\omega) = 1$ with high probability

- Unstructured search (db, key, etc.)
- **Classical solution:** $\mathcal{O}(N)$
- **Quantum solution:** $\mathcal{O}(\sqrt{N})$
- **Intuition:** divide security level by 2

Grover's Search

Grover's search

INPUT: $f : \{0, \ldots, N-1\} \to \{0, 1\}$ where $f(x) = 1$ for a single x

OUTPUT: ω such that $f(\omega) = 1$ with high probability

- Unstructured search (db, key, etc.)
- Classical solution: $\mathcal{O}(N)$
- Quantum solution: $\mathcal{O}(\sqrt{N})$
- **Intuition:** divide security level by 2

Grover's Search

Grover's search

INPUT: $f : \{0, \ldots, N-1\} \to \{0,1\}$ where $f(x) = 1$ for a single x

OUTPUT: ω such that $f(\omega) = 1$ with high probability

- Unstructured search (db, key, etc.)
- **Classical solution:** $\mathcal{O}(N)$
- **Quantum solution:** $\mathcal{O}(\sqrt{N})$
- **Intuition:** divide security level by 2

Grover's Search

Grover's search

INPUT: $f : \{0, \ldots, N-1\} \to \{0, 1\}$ where $f(x) = 1$ for a single x

OUTPUT: ω such that $f(\omega) = 1$ with high probability

- Unstructured search (db, key, etc.)
- **Classical solution:** $\mathcal{O}(N)$
- **Quantum solution:** $\mathcal{O}(\sqrt{N})$
- **Intuition:** divide security level by 2

Grover's Search

Grover's search

INPUT: $f : \{0, \ldots, N-1\} \to \{0,1\}$ where $f(x) = 1$ for a single x

OUTPUT: ω such that $f(\omega) = 1$ with high probability

- Unstructured search (db, key, etc.)
- **Classical solution:** $\mathcal{O}(N)$
- **Quantum solution:** $\mathcal{O}(\sqrt{N})$
- **Intuition:** divide security level by 2

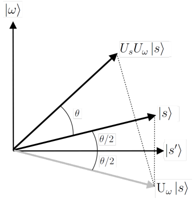
Grover's Search

Grover's search

INPUT: $f : \{0, \ldots, N-1\} \rightarrow \{0, 1\}$ where $f(x) = 1$ for a single x

OUTPUT: ω such that $f(\omega) = 1$ with high probability

- Unstructured search (db, key, etc.)
- **Classical solution:** $\mathcal{O}(N)$
- **Quantum solution:** $\mathcal{O}(\sqrt{N})$
- **Intuition:** divide security level by 2



(Source: Wikipedia)

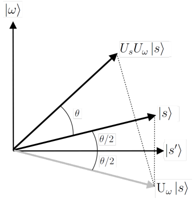
Grover's Search

Grover's search

INPUT: $f : \{0, \dots, N-1\} \to \{0, 1\}$ where $f(x) = 1$ for a single x

OUTPUT: ω such that $f(\omega) = 1$ with high probability

- Unstructured search (db, key, etc.)
- **Classical solution:** $\mathcal{O}(N)$
- **Quantum solution:** $\mathcal{O}(\sqrt{N})$
- **Intuition:** divide security level by 2



(Source: Wikipedia)

$$U_\omega = I - 2\,|\omega\rangle\langle\omega|$$

$$U_s = 2\,|s\rangle\langle s| - I$$

Grover's Search

Impact to cryptography

- **Symmetric-key encryption:** AES-128, AES-192, AES-256
  - brute-force key: $256 \rightarrow 128$,
    $192 \rightarrow 96$,
    $128 \rightarrow 64$
- **Cryptographic hash function:** SHA-256, SHA3-384, SHA3-512
  - preimage-resistence: $512 \rightarrow 256$,
    $384 \rightarrow 192$,
    $256 \rightarrow 128$
  - collision-resistance: $512 \rightarrow 170$,
    $384 \rightarrow 128$,
    $256 \rightarrow 85$

BHT algorithm (Brassard, Høyer, Tapp, 1998):

Combine **Grover's search** + **Birthday attack** to find collisions in $\mathcal{O}(2^{n/3})$

Grover's Search

Impact to cryptography

- **Symmetric-key encryption:** AES-128, AES-192, AES-256
  - brute-force key: $256 \to 128$, $192 \to 96$, $128 \to 64$
- **Cryptographic hash function:** SHA-256, SHA3-384, SHA3-512
  - preimage-resistence: $512 \to 256$, $384 \to 192$, $256 \to 128$
  - collision-resistance: $512 \to 170$, $384 \to 128$, $256 \to 85$

BHT algorithm (Brassard, Høyer, Tapp, 1998):

Combine **Grover's search** + **Birthday attack** to find collisions in $\mathcal{O}(2^{n/3})$

Grover's Search

Impact to cryptography

- **Symmetric-key encryption:** AES-128, AES-192, AES-256
  - brute-force key: $256 \to 128$,
    $192 \to 96$,
    $128 \to 64$
- **Cryptographic hash function:** SHA-256, SHA3-384, SHA3-512
  - preimage-resistence: $512 \to 256$,
    $384 \to 192$,
    $256 \to 128$
  - collision-resistance: $512 \to 170$,
    $384 \to 128$,
    $256 \to 85$

BHT algorithm (Brassard, Høyer, Tapp, 1998):

Combine **Grover's search** + **Birthday attack** to find collisions in $\mathcal{O}(2^{n/3})$

Grover's Search

Impact to cryptography

- **Symmetric-key encryption:** AES-128, AES-192, AES-256
  - brute-force key: $256 \to 128$, $192 \to 96$, $128 \to 64$
- **Cryptographic hash function:** SHA-256, SHA3-384, SHA3-512
  - preimage-resistence: $512 \to 256$, $384 \to 192$, $256 \to 128$
  - collision-resistance: $512 \to 170$, $384 \to 128$, $256 \to 85$

BHT algorithm (Brassard, Høyer, Tapp, 1998):

Combine **Grover's search** + **Birthday attack** to find collisions in $\mathcal{O}(2^{n/3})$

Grover's Search

Impact to cryptography

- **Symmetric-key encryption:** AES-128, AES-192, AES-256
  - brute-force key: $256 \rightarrow 128$,
    $192 \rightarrow 96$,
    $128 \rightarrow 64$
- **Cryptographic hash function:** SHA-256, SHA3-384, SHA3-512
  - preimage-resistence: $512 \rightarrow 256$,
    $384 \rightarrow 192$,
    $256 \rightarrow 128$
  - collision-resistance: $512 \rightarrow 170$,
    $384 \rightarrow 128$,
    $256 \rightarrow 85$

BHT algorithm (Brassard, Høyer, Tapp, 1998):

Combine **Grover's search** + **Birthday attack** to find collisions in $\mathcal{O}(2^{n/3})$

Grover's Search

Impact to cryptography

- **Symmetric-key encryption:** AES-128, AES-192, AES-256
  - brute-force key: $256 \to 128$, $192 \to 96$, $128 \to 64$
- **Cryptographic hash function:** SHA-256, SHA3-384, SHA3-512
  - preimage-resistence: $512 \to 256$, $384 \to 192$, $256 \to 128$
  - collision-resistance: $512 \to 170$, $384 \to 128$, $256 \to 85$

BHT algorithm (Brassard, Høyer, Tapp, 1998):

Combine **Grover's search** + **Birthday attack** to find collisions in $\mathcal{O}(2^{n/3})$

Shor's Algorithm
Introduction

- Shor (1994): factoring and discrete log are **easy for quantum computers**
- Massive impact to **public-key cryptography standards** (in use even today!)
- Changed the field of **secure communication** and **quantum computing**
- Ekerå and Gidney (2021) $\sim$**20 million qubits**, 8 hours to break RSA-2048
- Regev (2023), Ragavan & Vaikuntanathan (2024) make factoring **more practical**

Develop *quantum-safe cryptography*

Shor's Algorithm
Introduction

- Shor (1994): factoring and discrete log are **easy for quantum computers**
- Massive impact to **public-key cryptography standards** (in use even today!)
- Changed the field of **secure communication** and **quantum computing**
- Ekerå and Gidney (2021) $\sim$**20 million qubits**, 8 hours to break RSA-2048
- Regev (2023), Ragavan & Vaikuntanathan (2024) make factoring **more practical**

Develop *quantum-safe cryptography*

Shor's Algorithm

Introduction

- Shor (1994): factoring and discrete log are **easy for quantum computers**
- Massive impact to **public-key cryptography standards** (in use even today!)
- Changed the field of **secure communication** and **quantum computing**
- Ekerå and Gidney (2021) $\sim$**20 million qubits**, 8 hours to break RSA-2048
- Regev (2023), Ragavan & Vaikuntanathan (2024) make factoring **more practical**

Develop *quantum-safe cryptography*

Shor's Algorithm

Introduction

- Shor (1994): factoring and discrete log are **easy for quantum computers**
- Massive impact to **public-key cryptography standards** (in use even today!)
- Changed the field of **secure communication** and **quantum computing**
- Ekerå and Gidney (2021) $\sim$**20 million qubits**, 8 hours to break RSA-2048
- Regev (2023), Ragavan & Vaikuntanathan (2024) make factoring **more practical**

Develop *quantum-safe cryptography*

Shor's Algorithm

Introduction

- Shor (1994): factoring and discrete log are **easy for quantum computers**
- Massive impact to **public-key cryptography standards** (in use even today!)
- Changed the field of **secure communication** and **quantum computing**
- Ekerå and Gidney (2021) $\sim$**20 million qubits**, 8 hours to break RSA-2048
- Regev (2023), Ragavan & Vaikuntanathan (2024) make factoring **more practical**

Develop *quantum-safe cryptography*

Shor's Algorithm

Introduction

- Shor (1994): factoring and discrete log are **easy for quantum computers**
- Massive impact to **public-key cryptography standards** (in use even today!)
- Changed the field of **secure communication** and **quantum computing**
- Ekerå and Gidney (2021) $\sim$**20 million qubits**, 8 hours to break RSA-2048
- Regev (2023), Ragavan & Vaikuntanathan (2024) make factoring **more practical**

Develop *quantum-safe cryptography*

Shor's Algorithm

Intuition

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ f(x) = a^x \pmod N \end{cases}$$

1. Start with uniform superposition $|x\rangle = \frac{1}{2^{2n}} \sum_{s=0}^{2^{2n}-1} |s\rangle$ $(\mathcal{O}(n))$
2. Evaluate f on $|x\rangle$ $(\mathcal{O}(n^3))$
3. Preform Quantum Fourier Transform QFT$(f(|x\rangle))$ $(\mathcal{O}(n^2))$
4. Measure to read output, the period of f $(\mathcal{O}(n))$

Find period $\overset{\text{classical}}{\Longrightarrow}$ **Factor** N

- Classical complexity (GNFS): $\mathcal{O}(e^{1.9(n)^{1/3}\log(n)^{2/3}})$
- Quantum complexity (Shor): $\mathcal{O}(n^3)$

Shor's Algorithm

Intuition

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ f(x) = a^x \pmod N \end{cases}$$

1. Start with uniform superposition $|x\rangle = \frac{1}{2^{2n}} \sum_{s=0}^{2^{2n}-1} |s\rangle$ $(\mathcal{O}(n))$
2. Evaluate f on $|x\rangle$ $(\mathcal{O}(n^3))$
3. Preform Quantum Fourier Transform $\text{QFT}(f(|x\rangle))$ $(\mathcal{O}(n^2))$
4. Measure to read output, the period of f $(\mathcal{O}(n))$

Find period $\overset{\text{classical}}{\Longrightarrow}$ **Factor** N

- Classical complexity (GNFS): $\mathcal{O}(e^{1.9(n)^{1/3}\log(n)^{2/3}})$
- Quantum complexity (Shor): $\mathcal{O}(n^3)$

Shor's Algorithm

Intuition

$$\texttt{INPUT:}\begin{cases} N = p \cdot q \\ f(x) = a^x \pmod N \end{cases}$$

1. Start with uniform superposition $|x\rangle = \frac{1}{2^{2n}} \sum_{s=0}^{2^{2n}-1} |s\rangle$ $(\mathcal{O}(n))$
2. Evaluate f on $|x\rangle$ $(\mathcal{O}(n^3))$
3. Preform Quantum Fourier Transform QFT$(f(|x\rangle))$ $(\mathcal{O}(n^2))$
4. Measure to read output, the period of f $(\mathcal{O}(n))$

Find period $\overset{\text{classical}}{\Longrightarrow}$ **Factor** N

- Classical complexity (GNFS): $\mathcal{O}(e^{1.9(n)^{1/3}\log(n)^{2/3}})$
- Quantum complexity (Shor): $\mathcal{O}(n^3)$

Shor's Algorithm

Intuition

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ f(x) = a^x \pmod N \end{cases}$$

1. Start with uniform superposition $|x\rangle = \frac{1}{2^{2n}} \sum_{s=0}^{2^{2n}-1} |s\rangle$ $(\mathcal{O}(n))$
2. Evaluate f on $|x\rangle$ $(\mathcal{O}(n^3))$
3. Preform Quantum Fourier Transform $\mathrm{QFT}(f(|x\rangle))$ $(\mathcal{O}(n^2))$
4. Measure to read output, the period of f $(\mathcal{O}(n))$

Find period $\overset{\text{classical}}{\Longrightarrow}$ **Factor** N

- Classical complexity (GNFS): $\mathcal{O}(e^{1.9(n)^{1/3}\log(n)^{2/3}})$
- Quantum complexity (Shor): $\mathcal{O}(n^3)$

Shor's Algorithm

Intuition

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ f(x) = a^x \pmod{N} \end{cases}$$

1. Start with uniform superposition $|x\rangle = \frac{1}{2^{2n}} \sum_{s=0}^{2^{2n}-1} |s\rangle$ $\quad (\mathcal{O}(n))$
2. Evaluate f on $|x\rangle$ $\quad (\mathcal{O}(n^3))$
3. Preform Quantum Fourier Transform QFT$(f(|x\rangle))$ $\quad (\mathcal{O}(n^2))$
4. Measure to read output, the period of f $\quad (\mathcal{O}(n))$

Find period $\overset{\text{classical}}{\Longrightarrow}$ Factor N

- Classical complexity (GNFS): $\mathcal{O}(e^{1.9(n)^{1/3}\log(n)^{2/3}})$
- Quantum complexity (Shor): $\mathcal{O}(n^3)$

Shor's Algorithm

Intuition

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ f(x) = a^x \pmod N \end{cases}$$

1. Start with uniform superposition $|x\rangle = \frac{1}{2^{2n}} \sum_{s=0}^{2^{2n}-1} |s\rangle$ $(\mathcal{O}(n))$
2. Evaluate f on $|x\rangle$ $(\mathcal{O}(n^3))$
3. Preform Quantum Fourier Transform $\text{QFT}(f(|x\rangle))$ $(\mathcal{O}(n^2))$
4. Measure to read output, the period of f $(\mathcal{O}(n))$

Find period $\overset{\text{classical}}{\Longrightarrow}$ **Factor** N

- Classical complexity (GNFS): $\mathcal{O}(e^{1.9(n)^{1/3} \log(n)^{2/3}})$
- Quantum complexity (Shor): $\mathcal{O}(n^3)$

Shor's Algorithm

Intuition

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ f(x) = a^x \pmod{N} \end{cases}$$

1. Start with uniform superposition $|x\rangle = \frac{1}{2^{2n}} \sum_{s=0}^{2^{2n}-1} |s\rangle$ $(\mathcal{O}(n))$
2. Evaluate f on $|x\rangle$ $(\mathcal{O}(n^3))$
3. Preform Quantum Fourier Transform QFT$(f(|x\rangle))$ $(\mathcal{O}(n^2))$
4. Measure to read output, the period of f $(\mathcal{O}(n))$

Find period $\overset{\text{classical}}{\Longrightarrow}$ **Factor** N

- Classical complexity (GNFS): $\mathcal{O}(e^{1.9(n)^{1/3}\log(n)^{2/3}})$
- Quantum complexity (Shor): $\mathcal{O}(n^3)$

Shor's Algorithm

Classical reduction

$$\texttt{INPUT:} \begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod N) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod N \end{cases}$$

1. **if** $k = \gcd(a, N) \neq 1$, **return** $(k, N/k)$
2. $a^r = 1 \pmod N$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$ $(a^{r/2} \neq \pm 1)$
5. **return** $\left(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N)\right)$

Example

$$N = 15,\ a = 7,\ r = 4$$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm
Classical reduction

$$\texttt{INPUT:} \begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod N) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod N \end{cases}$$

1. **if** $k = \gcd(a, N) \neq 1$, **return** $(k, N/k)$
2. $a^r = 1 \pmod N$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so
   $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$
   $(a^{r/2} \neq \pm 1)$
5. **return** $(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N))$

Example

$N = 15,\ a = 7,\ r = 4$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm

Classical reduction

$$\texttt{INPUT:} \begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod N) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod N \end{cases}$$

1. **if** $k = \gcd(a, N) \neq 1$, **return** $(k, N/k)$
2. $a^r = 1 \pmod N$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$ $(a^{r/2} \neq \pm 1)$
5. **return** $(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N))$

Example

$N = 15,\ a = 7,\ r = 4$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm

Classical reduction

$$\texttt{INPUT:} \begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod N) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod N \end{cases}$$

1. $\texttt{if}$ $k = \gcd(a, N) \neq 1$, $\texttt{return}$ $(k, N/k)$
2. $a^r = 1 \pmod N$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so
   $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$
   $(a^{r/2} \neq \pm 1)$
5. $\texttt{return}$ $\left(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N)\right)$

Example

$N = 15$, $a = 7$, $r = 4$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm

Classical reduction

$$\texttt{INPUT:}\begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod N) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod N \end{cases}$$

1. **if** $k = \gcd(a, N) \neq 1$, **return** $(k, N/k)$
2. $a^r = 1 \pmod N$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$
   $(a^{r/2} \neq \pm 1)$
5. **return** $(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N))$

Example

$N = 15$, $a = 7$, $r = 4$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm

Classical reduction

$$\texttt{INPUT:} \begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod{N}) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod{N} \end{cases}$$

1. **if** $k = \gcd(a, N) \neq 1$, **return** $(k, N/k)$
2. $a^r = 1 \pmod{N}$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$ $(a^{r/2} \neq \pm 1)$
5. **return** $\left(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N)\right)$

Example

$N = 15,\ a = 7,\ r = 4$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm

Classical reduction

$$\texttt{INPUT:} \begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod N) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod N \end{cases}$$

1. **if** $k = \gcd(a, N) \neq 1$, **return** $(k, N/k)$
2. $a^r = 1 \pmod N$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so
   $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$
   $(a^{r/2} \neq \pm 1)$
5. **return** $\left(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N)\right)$

Example

$N = 15,\ a = 7,\ r = 4$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm

Classical reduction

$$\texttt{INPUT:}\begin{cases} a \\ r, \text{ order of } a \quad (\text{smallest } i,\ a^i = 1 \pmod N) \\ \quad r \text{ is even, } a^{r/2} \neq -1 \pmod N \end{cases}$$

1. **if** $k = \gcd(a, N) \neq 1$, **return** $(k, N/k)$
2. $a^r = 1 \pmod N$, so $N \mid (a^r - 1)$
3. $a^r - 1 = (a^{r/2} - 1)(a^{r/2} + 1)$, so
   $N \mid (a^{r/2} - 1)(a^{r/2} + 1)$
4. But $N \nmid (a^{r/2} - 1)$ and $N \nmid (a^{r/2} + 1)$
   ($a^{r/2} \neq \pm 1$)
5. **return** $\left(\gcd(a^{r/2} + 1, N), \gcd(a^{r/2} - 1, N)\right)$

Example

$$N = 15,\ a = 7,\ r = 4$$

$$\begin{cases} r \text{ is even} \\ 7^{4/2} = 4 \neq -1 \pmod{15} \end{cases}$$

$$p = \gcd(7^2 - 1, 15) = 3$$
$$q = \gcd(7^2 + 1, 15) = 5$$

$$N = 15 = 3 \cdot 5 = p \cdot q$$

Shor's Algorithm

Quantum Order Finding

INPUT: $\begin{cases} N = p \cdot q \\ U : |x\rangle |0\rangle \to |x\rangle |a^x \pmod N\rangle \end{cases}$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = (H^{\otimes t} \otimes I^{\otimes n}) |0\rangle^{\otimes t} |0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1} |x\rangle |0\rangle$
2. $|\psi_2\rangle = U |\psi_2\rangle = \sum_{x=0}^{2^t-1} |x\rangle |a^x \pmod N\rangle$
3. Measure $y_0 = a^{x_0} \pmod N$
   $|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle$
4. $|\psi_4\rangle = \text{QFT}_{2^t} |\psi_4\rangle =$
   $\frac{1}{\sqrt{m}} \frac{1}{\sqrt{2^t}} \sum_{y=0}^{2^t-1} \sum_{k=0}^{m-1} e^{2\pi i \frac{(x_0+kr)y}{2^t}} |y\rangle$
5. Measure $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$
   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \Longrightarrow$ Recover r

Example

$N = 15,\ a = 7,\ t = 8$

$|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |0\rangle$

$|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_x |x\rangle |7^x \pmod{15}\rangle$

$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + 4k\rangle$

$\sum_k e^{2\pi i \frac{4k}{2^t} y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \Longrightarrow r = 4$

Shor's Algorithm

Quantum Order Finding

INPUT: $\begin{cases} N = p \cdot q \\ U : |x\rangle |0\rangle \to |x\rangle |a^x \pmod N\rangle \end{cases}$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = \left(H^{\otimes t} \otimes I^{\otimes n}\right) |0\rangle^{\otimes t} |0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1} |x\rangle |0\rangle$
2. $|\psi_2\rangle = U |\psi_2\rangle = \sum_{x=0}^{2^t-1} |x\rangle |a^x \pmod N\rangle$
3. Measure $y_0 = a^{x_0} \pmod N$
   $|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle$
4. $|\psi_4\rangle = \text{QFT}_{2^t} |\psi_4\rangle =$
   $\frac{1}{\sqrt{m}} \frac{1}{\sqrt{2^t}} \sum_{y=0}^{2^t-1} \sum_{k=0}^{m-1} e^{2\pi i \frac{(x_0+kr)y}{2^t}} |y\rangle$
5. Measure $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$
   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \implies$ Recover r

Example

$N = 15$, $a = 7$, $t = 8$

$|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |0\rangle$

$|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_x |x\rangle |7^x \pmod{15}\rangle$

$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + 4k\rangle$

$\sum_k e^{2\pi i \frac{4k}{2^t} y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \implies r = 4$

Shor's Algorithm

Quantum Order Finding

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ U : |x\rangle |0\rangle \to |x\rangle |a^x \pmod{N}\rangle \end{cases}$$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = \left(H^{\otimes t} \otimes I^{\otimes n}\right) |0\rangle^{\otimes t} |0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1} |x\rangle |0\rangle$
2. $|\psi_2\rangle = U |\psi_2\rangle = \sum_{x=0}^{2^t-1} |x\rangle |a^x \pmod{N}\rangle$
3. Measure $y_0 = a^{x_0} \pmod{N}$
   $|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle$
4. $|\psi_4\rangle = \mathrm{QFT}_{2^t} |\psi_4\rangle =$
   $\frac{1}{\sqrt{m}} \frac{1}{\sqrt{2^t}} \sum_{y=0}^{2^t-1} \sum_{k=0}^{m-1} e^{2\pi i \frac{(x_0+kr)y}{2^t}} |y\rangle$
5. Measure $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$
   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \implies$ Recover r

Example

$N = 15$, $a = 7$, $t = 8$

$|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |0\rangle$

$|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_x |x\rangle |7^x \pmod{15}\rangle$

$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + 4k\rangle$

$\sum_k e^{2\pi i \frac{4k}{2^t} y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \implies r = 4$

Shor's Algorithm

Quantum Order Finding

INPUT: $\begin{cases} N = p \cdot q \\ U : |x\rangle |0\rangle \to |x\rangle |a^x \pmod N\rangle \end{cases}$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = \left(H^{\otimes t} \otimes I^{\otimes n}\right) |0\rangle^{\otimes t} |0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1} |x\rangle |0\rangle$
2. $|\psi_2\rangle = U |\psi_2\rangle = \sum_{x=0}^{2^t-1} |x\rangle |a^x \pmod N\rangle$
3. **Measure** $y_0 = a^{x_0} \pmod N$

   $|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle$
4. $|\psi_4\rangle = \text{QFT}_{2^t} |\psi_4\rangle =$

   $\frac{1}{\sqrt{m}} \frac{1}{\sqrt{2^t}} \sum_{y=0}^{2^t-1} \sum_{k=0}^{m-1} e^{2\pi i \frac{(x_0+kr)y}{2^t}} |y\rangle$
5. **Measure** $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$

   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \implies$ Recover r

Example

$N = 15,\ a = 7,\ t = 8$

$|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |0\rangle$

$|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_x |x\rangle |7^x \pmod{15}\rangle$

$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + 4k\rangle$

$\sum_k e^{2\pi i \frac{4k}{2^t} y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \implies r = 4$

Shor's Algorithm

Quantum Order Finding

INPUT: $\begin{cases} N = p \cdot q \\ U : |x\rangle |0\rangle \to |x\rangle |a^x \pmod N\rangle \end{cases}$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = \left(H^{\otimes t} \otimes I^{\otimes n}\right) |0\rangle^{\otimes t} |0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1} |x\rangle |0\rangle$
2. $|\psi_2\rangle = U |\psi_2\rangle = \sum_{x=0}^{2^t-1} |x\rangle |a^x \pmod N\rangle$
3. Measure $y_0 = a^{x_0} \pmod N$
   $|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle$
4. $|\psi_4\rangle = \mathsf{QFT}_{2^t} |\psi_4\rangle =$
   $\frac{1}{\sqrt{m}} \frac{1}{\sqrt{2^t}} \sum_{y=0}^{2^t-1} \sum_{k=0}^{m-1} e^{2\pi i \frac{(x_0+kr)y}{2^t}} |y\rangle$
5. Measure $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$
   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \implies$ Recover r

Example

$N = 15$, $a = 7$, $t = 8$

$|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |0\rangle$

$|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_x |x\rangle |7^x \pmod{15}\rangle$

$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + 4k\rangle$

$\sum_k e^{2\pi i \frac{4k}{2^t} y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \implies r = 4$

Shor's Algorithm

Quantum Order Finding

INPUT: $\begin{cases} N = p \cdot q \\ U : |x\rangle|0\rangle \to |x\rangle|a^x \pmod N\rangle \end{cases}$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = (H^{\otimes t} \otimes I^{\otimes n})|0\rangle^{\otimes t}|0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1}|x\rangle|0\rangle$
2. $|\psi_2\rangle = U|\psi_2\rangle = \sum_{x=0}^{2^t-1}|x\rangle|a^x \pmod N\rangle$
3. Measure $y_0 = a^{x_0} \pmod N$
   $|\psi_3\rangle = \frac{1}{\sqrt{m}}\sum_{k=0}^{m-1}|x_0 + kr\rangle$
4. $|\psi_4\rangle = \mathrm{QFT}_{2^t}|\psi_4\rangle =$
   $\frac{1}{\sqrt{m}}\frac{1}{\sqrt{2^t}}\sum_{y=0}^{2^t-1}\sum_{k=0}^{m-1} e^{2\pi i\frac{(x_0+kr)y}{2^t}}|y\rangle$
5. Measure $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$
   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \implies$ Recover r

Example

$N = 15,\ a = 7,\ t = 8$

$|\psi_1\rangle = \frac{1}{\sqrt{256}}\sum_{x=0}^{255}|x\rangle|0\rangle$

$|\psi_2\rangle = \frac{1}{\sqrt{256}}\sum_x|x\rangle|7^x \pmod{15}\rangle$

$|\psi_3\rangle = \frac{1}{\sqrt{m}}\sum_{k=0}^{m-1}|x_0 + 4k\rangle$

$\sum_k e^{2\pi i\frac{4k}{2^t}y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \implies r = 4$

Shor's Algorithm

Quantum Order Finding

INPUT: $\begin{cases} N = p \cdot q \\ U : |x\rangle |0\rangle \to |x\rangle |a^x \pmod{N}\rangle \end{cases}$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = \left(H^{\otimes t} \otimes I^{\otimes n}\right) |0\rangle^{\otimes t} |0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1} |x\rangle |0\rangle$
2. $|\psi_2\rangle = U |\psi_2\rangle = \sum_{x=0}^{2^t-1} |x\rangle |a^x \pmod{N}\rangle$
3. Measure $y_0 = a^{x_0} \pmod{N}$
   $|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle$
4. $|\psi_4\rangle = \mathsf{QFT}_{2^t} |\psi_4\rangle =$
   $\frac{1}{\sqrt{m}} \frac{1}{\sqrt{2^t}} \sum_{y=0}^{2^t-1} \sum_{k=0}^{m-1} e^{2\pi i \frac{(x_0+kr)y}{2^t}} |y\rangle$
5. Measure $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$
   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \implies$ Recover r

Example

$N = 15$, $a = 7$, $t = 8$

$|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle |0\rangle$

$|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_x |x\rangle |7^x \pmod{15}\rangle$

$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + 4k\rangle$

$\sum_k e^{2\pi i \frac{4k}{2^t} y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \implies r = 4$

Shor's Algorithm

Quantum Order Finding

$$\texttt{INPUT:} \begin{cases} N = p \cdot q \\ U : |x\rangle\,|0\rangle \to |x\rangle\,|a^x \pmod N\rangle \end{cases}$$

0. $t \propto \lceil \log_2 N^2 \rceil$, $n = \lceil \log_2 N \rceil$
1. $|\psi_1\rangle = \left(H^{\otimes t} \otimes I^{\otimes n}\right) |0\rangle^{\otimes t} |0\rangle^{\otimes n} = \sum_{x=0}^{2^t-1} |x\rangle\,|0\rangle$
2. $|\psi_2\rangle = U\,|\psi_2\rangle = \sum_{x=0}^{2^t-1} |x\rangle\,|a^x \pmod N\rangle$
3. Measure $y_0 = a^{x_0} \pmod N$
   $|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle$
4. $|\psi_4\rangle = \mathsf{QFT}_{2^t}\,|\psi_4\rangle =$
   $\frac{1}{\sqrt{m}} \frac{1}{\sqrt{2^t}} \sum_{y=0}^{2^t-1} \sum_{k=0}^{m-1} e^{2\pi i \frac{(x_0+kr)y}{2^t}}\,|y\rangle$
5. Measure $y \approx s\frac{2^t}{r}$ $(s \in \mathbb{Z})$
   $\frac{s}{r}$ is a convergent of $\frac{y}{2^t} \Longrightarrow$ Recover r

Example

$$N = 15,\; a = 7,\; t = 8$$

$$|\psi_1\rangle = \frac{1}{\sqrt{256}} \sum_{x=0}^{255} |x\rangle\,|0\rangle$$

$$|\psi_2\rangle = \frac{1}{\sqrt{256}} \sum_{x} |x\rangle\,|7^x \pmod{15}\rangle$$

$$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + 4k\rangle$$

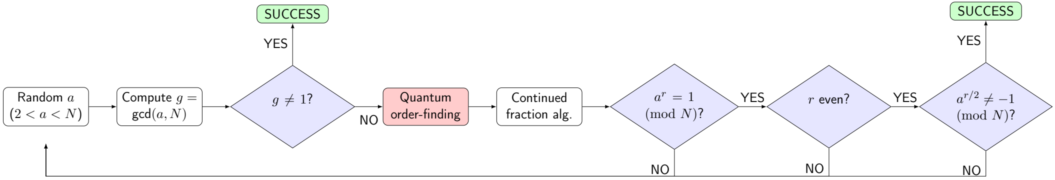
$\sum_k e^{2\pi i \frac{4k}{2^t} y}$ peaks at $y \approx s\frac{2^t}{4}$

Measure $y = 64$

$64/256 = 1/4 \Longrightarrow r = 4$

Shor's Algorithm

Review

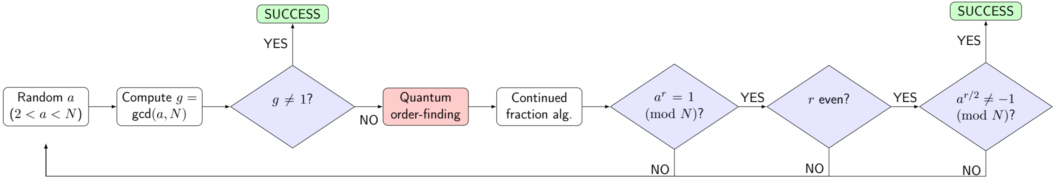


Missing:

- Efficient to find *good* a with high probability
- gcd and continued fractions algorithms are efficient
- Modular exponentiation implementation (Oracle U)
- Probability peak separation, i.e. measuring gives a close-enough y

Shor's Algorithm

Review

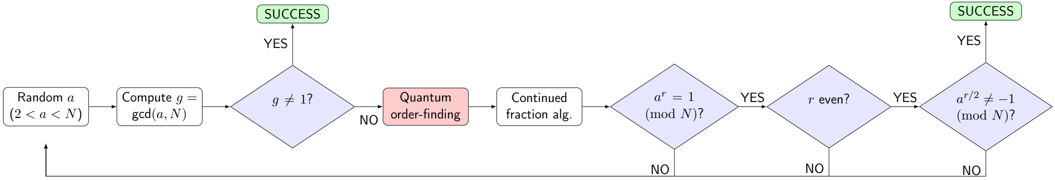


Missing:

- Efficient to find *good* a with high probability
- gcd and continued fractions algorithms are efficient
- Modular exponentiation implementation (Oracle U)
- Probability peak separation, i.e. measuring gives a close-enough y

Shor's Algorithm

Review

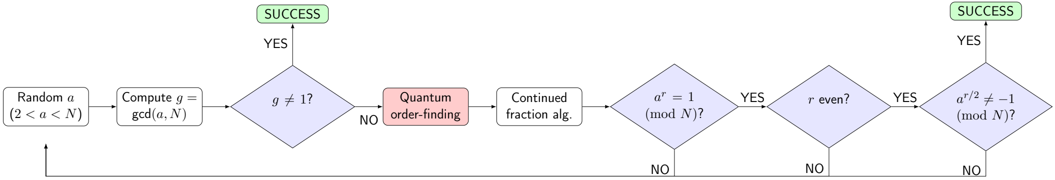


Missing:

- Efficient to find *good* a with high probability
- gcd and continued fractions algorithms are efficient
- Modular exponentiation implementation (Oracle U)
- Probability peak separation, i.e. measuring gives a close-enough y

Shor's Algorithm

Review

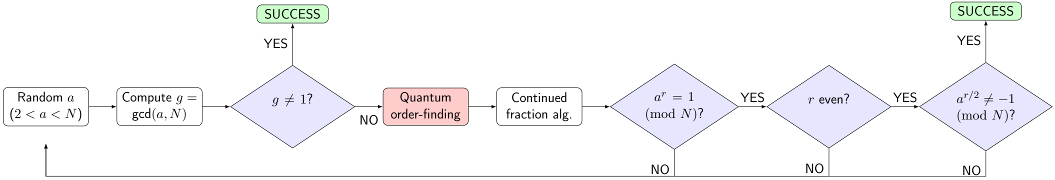


Missing:

- Efficient to find *good* a with high probability
- gcd and continued fractions algorithms are efficient
- Modular exponentiation implementation (Oracle U)
- Probability peak separation, i.e. measuring gives a close-enough y

Shor's Algorithm

Review

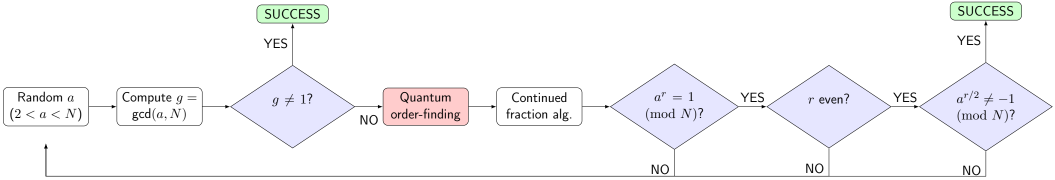


Missing:

- Efficient to find *good* a with high probability
- gcd and continued fractions algorithms are efficient
- Modular exponentiation implementation (Oracle U)
- Probability peak separation, i.e. measuring gives a close-enough y

Shor's Algorithm

Review



Missing:

- Efficient to find *good* a with high probability
- gcd and $\mathrm{continued\ fractions}$ algorithms are efficient
- $\mathrm{Modular\ exponentiation}$ implementation (Oracle U)
- Probability peak separation, i.e. measuring gives a close-enough y

Hidden Subgroup Problem

Intuition

Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod{N}$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

Hidden Subgroup Problem

Intuition

Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod{N}$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

Hidden Subgroup Problem

Intuition

Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod N$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

Hidden Subgroup Problem

Intuition

Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod{N}$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

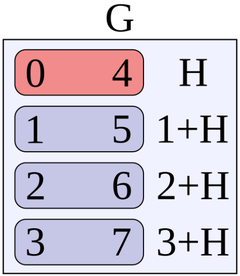
Hidden Subgroup Problem

Intuition

Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

$$G = (\mathbb{Z}_8, +)$$
$$f(x) = x \mod 4$$



(Source: Wikipedia)

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod{N}$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

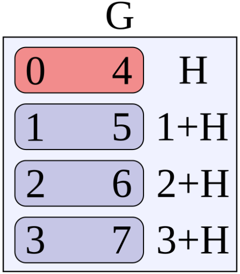
Hidden Subgroup Problem

Intuition

Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

$$G = (\mathbb{Z}_8, +)$$
$$f(x) = x \mod 4$$



(Source: Wikipedia)

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod N$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

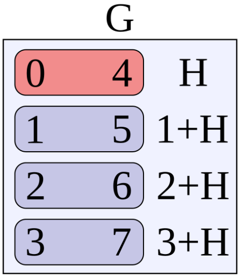
Hidden Subgroup Problem

Intuition

Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

$$G = (\mathbb{Z}_8, +)$$
$$f(x) = x \mod 4$$



(Source: Wikipedia)

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod{N}$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

Hidden Subgroup Problem

Intuition

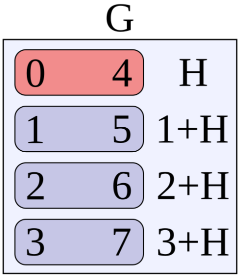
Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod N$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

$$G = (\mathbb{Z}_8, +)$$
$$f(x) = x \mod 4$$



(Source: Wikipedia)

Hidden Subgroup Problem

Intuition

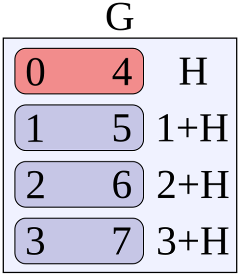
Find *hidden periodic* structure in group

- Given group G and function $f : G \to X$
- f is constant on cosets of unknown subgroup $H \subseteq G$
- **Goal:** Determine subgroup H

Connection to Shor's algorithm

- $G = \mathbb{Z}_{N^2}$ (integers modulo N^2)
- $f(x) = a^x \pmod N$ (modular exponentiation)
- Hidden subgroup $H = r\mathbb{Z}_{N^2}$ (r is the order of a)

$$G = (\mathbb{Z}_8, +)$$
$$f(x) = x \mod 4$$



(Source: Wikipedia)

Hidden Subgroup Problem

Problem	Quantum algorithm	Abelian?	Polynomial time?
Deutsch's problem	Deutsch's/Deutsch-Jozsa algorithm	Yes	Yes
Simon's problem	Simon's algorithm	Yes	Yes
Order finding	Shor's order finding algorithm	Yes	Yes
Discrete logarithm	Shor's algorithm for discrete logarithms	Yes	Yes
Period finding	Shor's algorithm	Yes	Yes
Abelian stabilizer	Kitaev's algorithm	Yes	Yes
Graph Isomorphism	None	No	No
Shortest vector problem	None	No	No

List of HSP quantum algorithms. (Source: Wikipedia)

Contents

Reversible Computation

- Classical computation often erases information — **irreversible**
- Quantum evolution is unitary (also, no fanout) — **reversible**
- **Compile** any irreversible $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^m$ into a reversible $\overline{f} : \mathbb{Z}_2^{n+m} \to \mathbb{Z}_2^{n+m}$

$$\overline{f}((x, y)) = (x, y \oplus f(x))$$

Example — AND

$$\mathsf{AND}(a, b) = a \wedge b$$

$$\downarrow$$

$$\overline{\mathsf{AND}}((a, b), 0) = ((a, b), a \wedge b)$$

Reversible Computation

- Classical computation often erases information — **irreversible**
- Quantum evolution is unitary (also, no fanout) — **reversible**
- **Compile** any irreversible $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^m$ into a reversible $\overline{f} : \mathbb{Z}_2^{n+m} \to \mathbb{Z}_2^{n+m}$

$$\overline{f}((x, y)) = (x, y \oplus f(x))$$

Example — AND

$$\mathsf{AND}(a, b) = a \wedge b$$

$$\downarrow$$

$$\overline{\mathsf{AND}}((a, b), 0) = ((a, b), a \wedge b)$$

Reversible Computation

- Classical computation often erases information — **irreversible**
- Quantum evolution is unitary (also, no fanout) — **reversible**
- **Compile** any irreversible $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^m$ into a reversible $\overline{f} : \mathbb{Z}_2^{n+m} \to \mathbb{Z}_2^{n+m}$

$$\overline{f}((x, y)) = (x, y \oplus f(x))$$

Example — AND

$$\text{AND}(a, b) = a \wedge b$$

$$\downarrow$$

$$\overline{\text{AND}}((a, b), 0) = ((a, b), a \wedge b)$$

Reversible Computation

- Classical computation often erases information — **irreversible**
- Quantum evolution is unitary (also, no fanout) — **reversible**
- **Compile** any irreversible $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^m$ into a reversible $\overline{f} : \mathbb{Z}_2^{n+m} \to \mathbb{Z}_2^{n+m}$

$$\overline{f}((x, y)) = (x, y \oplus f(x))$$

Example — AND

$$\mathsf{AND}(a, b) = a \wedge b$$
$$\downarrow$$
$$\overline{\mathsf{AND}}((a, b), 0) = ((a, b), a \wedge b)$$

Quantum Function Evaluation

- Quantum **circuit** implements unitary operator U acting on a state $U|\psi\rangle$
- Implement any classical function f as unitary operator $U_f : \mathbb{C}^{n+m} \to \mathbb{C}^{n+m}$

$$U_f |x\rangle |y\rangle \to |x\rangle |y \oplus f(x)\rangle$$

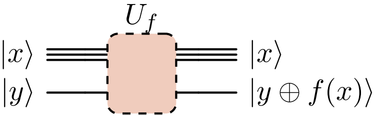
$|x\rangle, |y\rangle$ are **quantum states** — U_f acts on a ***superposition*** of inputs

$$U_f \sum_i \alpha_i |x_i\rangle |y\rangle = \sum_i \alpha_i |x_i\rangle |y \oplus f(x_i)\rangle$$

Quantum Function Evaluation

- Quantum **circuit** implements unitary operator U acting on a state $U|\psi\rangle$
- Implement any classical function f as unitary operator $U_f : \mathbb{C}^{n+m} \to \mathbb{C}^{n+m}$

$$U_f |x\rangle |y\rangle \to |x\rangle |y \oplus f(x)\rangle$$



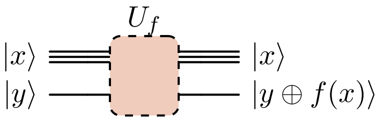
$|x\rangle, |y\rangle$ are **quantum states** — U_f acts on a ***superposition*** of inputs

$$U_f \sum_i \alpha_i |x_i\rangle |y\rangle = \sum_i \alpha_i |x_i\rangle |y \oplus f(x_i)\rangle$$

Quantum Function Evaluation

- Quantum **circuit** implements unitary operator U acting on a state $U|\psi\rangle$
- Implement any classical function f as unitary operator $U_f : \mathbb{C}^{n+m} \to \mathbb{C}^{n+m}$

$$U_f |x\rangle |y\rangle \to |x\rangle |y \oplus f(x)\rangle$$



$|x\rangle, |y\rangle$ are **quantum states** — U_f acts on a ***superposition*** of inputs

$$U_f \sum_i \alpha_i |x_i\rangle |y\rangle = \sum_i \alpha_i |x_i\rangle |y \oplus f(x_i)\rangle$$

Quantum Phase Oracle

Output register $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ may also be in a superposition

$$U_f \sum_i \alpha_i |x_i\rangle |y\rangle = U_f \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right)$$

$$= \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}}(|0 \oplus f(x_i)\rangle - |1 \oplus f(x_i)\rangle) \right)$$

$$= \sum_i (-1)^{f(x_i)} \alpha_i |x_i\rangle |y\rangle$$ ($|y\rangle$ is separable, often ommited)

Exploit interference pattern of f controlled by x_i

Quantum Phase Oracle

Output register $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ may also be in a superposition

$$U_f \sum_i \alpha_i |x_i\rangle |y\rangle = U_f \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right)$$

$$= \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}}(|0 \oplus f(x_i)\rangle - |1 \oplus f(x_i)\rangle) \right)$$

$$= \sum_i (-1)^{f(x_i)} \alpha_i |x_i\rangle |y\rangle$$ ($|y\rangle$ is separable, often ommited)

Exploit interference pattern of f controlled by x_i

Quantum Phase Oracle

Output register $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ may also be in a superposition

$$U_f \sum_i \alpha_i |x_i\rangle |y\rangle = U_f \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right)$$

$$= \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}} (|0 \oplus f(x_i)\rangle - |1 \oplus f(x_i)\rangle) \right)$$

$$= \sum_i (-1)^{f(x_i)} \alpha_i |x_i\rangle |y\rangle$$ ($|y\rangle$ is separable, often ommited)

Exploit interference pattern of f controlled by x_i

Quantum Phase Oracle

Output register $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ may also be in a superposition

$$
\begin{aligned}
U_f \sum_i \alpha_i |x_i\rangle |y\rangle &= U_f \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right) \\
&= \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}}(|0 \oplus f(x_i)\rangle - |1 \oplus f(x_i)\rangle) \right) \\
&= \sum_i (-1)^{f(x_i)} \alpha_i |x_i\rangle |y\rangle \quad \text{(}|y\rangle \text{ is separable, often ommited)}
\end{aligned}
$$

Exploit interference pattern of f controlled by x_i

Quantum Phase Oracle

Output register $|y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ may also be in a superposition

$$\begin{aligned} U_f \sum_i \alpha_i |x_i\rangle |y\rangle &= U_f \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \\ &= \sum_i \alpha_i |x_i\rangle \left(\frac{1}{\sqrt{2}} (|0 \oplus f(x_i)\rangle - |1 \oplus f(x_i))\rangle \right) \\ &= \sum_i (-1)^{f(x_i)} \alpha_i |x_i\rangle |y\rangle \quad \text{(}|y\rangle \text{ is separable, often ommited)} \end{aligned}$$

Exploit interference pattern of f controlled by x_i

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

let $L = \{\}$
for $\mathbf{x}_i \in \mathbb{Z}_2^n$
  let $\mathbf{y}_i = f(\mathbf{x}_i)$
  if $(\mathbf{x}_j, \mathbf{y}_j) \in L$ st $\mathbf{y}_i = \mathbf{y}_j$
    return $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
  else
    append $(\mathbf{x}_i, \mathbf{y}_i)$ to L
return $\mathbf{s} = 0$

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

1. `let` $L = \{\}$
2. `for` $\mathbf{x}_i \in \mathbb{Z}_2^n$
3. `let` $\mathbf{y}_i = f(\mathbf{x}_i)$
4. `if` $(\mathbf{x}_j, \mathbf{y}_j) \in L$ `st` $\mathbf{y}_i = \mathbf{y}_j$
5. `return` $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
6. `else`
7. `append` $(\mathbf{x}_i, \mathbf{y}_i)$ `to` L
8. `return` $\mathbf{s} = 0$

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

1. let $L = \{\}$
2. for $\mathbf{x}_i \in \mathbb{Z}_2^n$
3.   let $\mathbf{y}_i = f(\mathbf{x}_i)$
4.   if $(\mathbf{x}_j, \mathbf{y}_j) \in L$ st $\mathbf{y}_i = \mathbf{y}_j$
5.   return $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
6.   else
7.   append $(\mathbf{x}_i, \mathbf{y}_i)$ to L
8. return $\mathbf{s} = 0$

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

1. let $L = \{\}$
2. for $\mathbf{x}_i \in \mathbb{Z}_2^n$
3. let $\mathbf{y}_i = f(\mathbf{x}_i)$
4. if $(\mathbf{x}_j, \mathbf{y}_j) \in L$ st $\mathbf{y}_i = \mathbf{y}_j$
5. return $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
6. else
7. append $(\mathbf{x}_i, \mathbf{y}_i)$ to L
8. return $\mathbf{s} = 0$

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

let $L = \{\}$
for $\mathbf{x}_i \in \mathbb{Z}_2^n$
  let $\mathbf{y}_i = f(\mathbf{x}_i)$
  if $(\mathbf{x}_j, \mathbf{y}_j) \in L$ st $\mathbf{y}_i = \mathbf{y}_j$
    return $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
  else
    append $(\mathbf{x}_i, \mathbf{y}_i)$ to L
return $\mathbf{s} = 0$

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

let $L = \{\}$
for $\mathbf{x}_i \in \mathbb{Z}_2^n$
    let $\mathbf{y}_i = f(\mathbf{x}_i)$
    if $(\mathbf{x}_j, \mathbf{y}_j) \in L$ st $\mathbf{y}_i = \mathbf{y}_j$
        return $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
    else
        append $(\mathbf{x}_i, \mathbf{y}_i)$ to L
return $\mathbf{s} = 0$

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

1. `let` $L = \{\}$
2. `for` $\mathbf{x}_i \in \mathbb{Z}_2^n$
3. `let` $\mathbf{y}_i = f(\mathbf{x}_i)$
4. `if` $(\mathbf{x}_j, \mathbf{y}_j) \in L$ `st` $\mathbf{y}_i = \mathbf{y}_j$
5. `return` $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
6. `else`
7. `append` $(\mathbf{x}_i, \mathbf{y}_i)$ `to` L
8. `return` $\mathbf{s} = 0$

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

1. `let` $L = \{\}$
2. `for` $\mathbf{x}_i \in \mathbb{Z}_2^n$
3. `let` $\mathbf{y}_i = f(\mathbf{x}_i)$
4. `if` $(\mathbf{x}_j, \mathbf{y}_j) \in L$ `st` $\mathbf{y}_i = \mathbf{y}_j$
5. `return` $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
6. `else`
7. `append` $(\mathbf{x}_i, \mathbf{y}_i)$ `to` L
8. `return` $\mathbf{s} = 0$

Quantum Oracle Problems

Simon's Problem

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical solution:

let $L = \{\}$
for $\mathbf{x}_i \in \mathbb{Z}_2^n$
    let $\mathbf{y}_i = f(\mathbf{x}_i)$
    if $(\mathbf{x}_j, \mathbf{y}_j) \in L$ st $\mathbf{y}_i = \mathbf{y}_j$
        return $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
    else
        append $(\mathbf{x}_i, \mathbf{y}_i)$ to L
return $\mathbf{s} = 0$

Quantum Oracle Problems

Simon's Problem

Simon's problem

Given: $f_{\mathbf{s}} : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$

Promise: $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{x} \oplus \mathbf{s})$

Goal: $\mathbf{s} \in \mathbb{Z}_2^n$

Classical: 2^n queries

Quantum: αn queries

Classical solution:

1. let $L = \{\}$
2. for $\mathbf{x}_i \in \mathbb{Z}_2^n$
3. let $\mathbf{y}_i = f(\mathbf{x}_i)$
4. if $(\mathbf{x}_j, \mathbf{y}_j) \in L$ st $\mathbf{y}_i = \mathbf{y}_j$
5. return $\mathbf{s} = \mathbf{x}_i \oplus \mathbf{x}_j$
6. else
7. append $(\mathbf{x}_i, \mathbf{y}_i)$ to L
8. return $\mathbf{s} = 0$

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n}) |\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}} (|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n} |\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} \left((-1)^{\mathbf{x}_0 \cdot \mathbf{y}} + (-1)^{(\mathbf{x}_0 \oplus \mathbf{s}) \cdot \mathbf{y}}\right) |\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} (-1)^{\mathbf{x}_0 \cdot \mathbf{y}} \underbrace{\left(1 + (-1)^{\mathbf{s} \cdot \mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s} \cdot \mathbf{y} = \mathbf{0}} |\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n}|0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n})|\psi_0\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f|\psi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}}(|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n}|\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}\left((-1)^{\mathbf{x}_0\cdot\mathbf{y}} + (-1)^{(\mathbf{x}_0\oplus\mathbf{s})\cdot\mathbf{y}}\right)|\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}(-1)^{\mathbf{x}_0\cdot\mathbf{y}}\underbrace{\left(1 + (-1)^{\mathbf{s}\cdot\mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s}\cdot\mathbf{y}=\mathbf{0}}|\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n}) |\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}} (|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n} |\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} \left((-1)^{\mathbf{x}_0 \cdot \mathbf{y}} + (-1)^{(\mathbf{x}_0 \oplus \mathbf{s}) \cdot \mathbf{y}}\right) |\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} (-1)^{\mathbf{x}_0 \cdot \mathbf{y}} \underbrace{(1 + (-1)^{\mathbf{s} \cdot \mathbf{y}})}_{\neq 0 \text{ if } \mathbf{s} \cdot \mathbf{y} = \mathbf{0}} |\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$

1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n}) |\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |0\rangle$

2. $|\psi_2\rangle = \mathcal{O}_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |f(\mathbf{x})\rangle$

3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}} (|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$

4. $|\psi_3\rangle = H^{\otimes n} |\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} \left((-1)^{\mathbf{x}_0 \cdot \mathbf{y}} + (-1)^{(\mathbf{x}_0 \oplus \mathbf{s}) \cdot \mathbf{y}}\right) |\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} (-1)^{\mathbf{x}_0 \cdot \mathbf{y}} \underbrace{\left(1 + (-1)^{\mathbf{s} \cdot \mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s} \cdot \mathbf{y} = \mathbf{0}} |\mathbf{y}\rangle$

5. `measure` $\mathbf{y}_i \perp \mathbf{s}$

6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n}) |\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$
   $|\psi_3\rangle = \frac{1}{\sqrt{2}} (|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n} |\psi_2\rangle$
   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} \left((-1)^{\mathbf{x}_0 \cdot \mathbf{y}} + (-1)^{(\mathbf{x}_0 \oplus \mathbf{s}) \cdot \mathbf{y}}\right) |\mathbf{y}\rangle$
   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} (-1)^{\mathbf{x}_0 \cdot \mathbf{y}} \underbrace{\left(1 + (-1)^{\mathbf{s} \cdot \mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s} \cdot \mathbf{y} = \mathbf{0}} |\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n}) |\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}} (|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n} |\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} \left((-1)^{\mathbf{x}_0 \cdot \mathbf{y}} + (-1)^{(\mathbf{x}_0 \oplus \mathbf{s}) \cdot \mathbf{y}}\right) |\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} (-1)^{\mathbf{x}_0 \cdot \mathbf{y}} \underbrace{\left(1 + (-1)^{\mathbf{s} \cdot \mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s} \cdot \mathbf{y} = 0} |\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n}|0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n})|\psi_0\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f|\psi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}}(|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n}|\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}\left((-1)^{\mathbf{x}_0\cdot\mathbf{y}} + (-1)^{(\mathbf{x}_0\oplus\mathbf{s})\cdot\mathbf{y}}\right)|\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}(-1)^{\mathbf{x}_0\cdot\mathbf{y}}\underbrace{\left(1 + (-1)^{\mathbf{s}\cdot\mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s}\cdot\mathbf{y}=\mathbf{0}}|\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n}) |\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}} (|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n} |\psi_2\rangle$

   $$= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} \left((-1)^{\mathbf{x}_0 \cdot \mathbf{y}} + (-1)^{(\mathbf{x}_0 \oplus \mathbf{s}) \cdot \mathbf{y}}\right) |\mathbf{y}\rangle$$

   $$= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} (-1)^{\mathbf{x}_0 \cdot \mathbf{y}} \underbrace{\left(1 + (-1)^{\mathbf{s} \cdot \mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s} \cdot \mathbf{y} = \mathbf{0}} |\mathbf{y}\rangle$$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n}|0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n})|\psi_0\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f|\psi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}}(|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n}|\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}\left((-1)^{\mathbf{x}_0\cdot\mathbf{y}} + (-1)^{(\mathbf{x}_0\oplus\mathbf{s})\cdot\mathbf{y}}\right)|\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}(-1)^{\mathbf{x}_0\cdot\mathbf{y}}\underbrace{\left(1+(-1)^{\mathbf{s}\cdot\mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s}\cdot\mathbf{y}=\mathbf{0}}|\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n}) |\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f |\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}} |\mathbf{x}\rangle |f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}} (|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n} |\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} \left((-1)^{\mathbf{x}_0 \cdot \mathbf{y}} + (-1)^{(\mathbf{x}_0 \oplus \mathbf{s}) \cdot \mathbf{y}}\right) |\mathbf{y}\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}} \sum_{\mathbf{y}} (-1)^{\mathbf{x}_0 \cdot \mathbf{y}} \underbrace{\left(1 + (-1)^{\mathbf{s} \cdot \mathbf{y}}\right)}_{\neq 0 \text{ if } \mathbf{s} \cdot \mathbf{y} = \mathbf{0}} |\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`

Quantum Oracle Problems

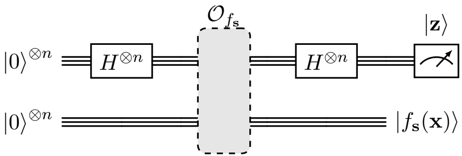
Simon's Problem (quantum solution)

0. $|\psi_0\rangle = |0\rangle^{\otimes n}|0\rangle^{\otimes n}$
1. $|\psi_1\rangle = (H^{\otimes n} \otimes I^{\otimes n})|\psi_0\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|0\rangle$
2. $|\psi_2\rangle = \mathcal{O}_f|\psi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}}|\mathbf{x}\rangle|f(\mathbf{x})\rangle$
3. `measure` $\mathbf{y}_0 = f(\mathbf{x}_0)$

   $|\psi_3\rangle = \frac{1}{\sqrt{2}}(|\mathbf{x}_0\rangle + |\mathbf{x}_0 \oplus \mathbf{s}\rangle)$
4. $|\psi_3\rangle = H^{\otimes n}|\psi_2\rangle$

   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}\left((-1)^{\mathbf{x}_0\cdot\mathbf{y}} + (-1)^{(\mathbf{x}_0\oplus\mathbf{s})\cdot\mathbf{y}}\right)|\mathbf{y}\rangle$

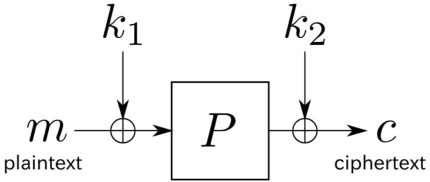
   $= \frac{1}{\sqrt{2^{n+1}}}\sum_{\mathbf{y}}(-1)^{\mathbf{x}_0\cdot\mathbf{y}}\underbrace{(1 + (-1)^{\mathbf{s}\cdot\mathbf{y}})}_{\neq 0 \text{ if } \mathbf{s}\cdot\mathbf{y}=\mathbf{0}}|\mathbf{y}\rangle$
5. `measure` $\mathbf{y}_i \perp \mathbf{s}$
6. `repeat until` n L.I. $\mathbf{y}_i$; `solve system`



Superposition Attacks

Block Ciphers

1. First proposal [KM10] — 3-round Feistel distinguisher
2. Even-Mansour key-recovery [KM12]:



The EM cipher

[KM12]

$$\mathsf{Enc}_{k_1,k_2}(m) = P(m \oplus k_1) \oplus k_2$$

$$f_{k_1}(m) = \mathsf{Enc}_{k_1,k_2}(m) \oplus P(m)$$

$$f_{k_1}(m) = P(m \oplus k_1) \oplus k_2 \oplus P(m)$$

$$f_{k_1}(m) = f(m \oplus k_1)$$

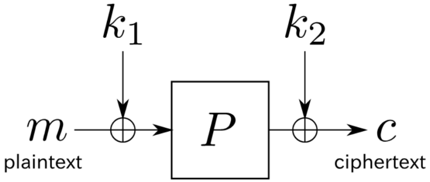
Simon's algorithm to recover k_1

$$k_2 = \mathsf{Enc}_{k_1,k_2}(m) \oplus P(m \oplus k_1)$$

Superposition Attacks

Block Ciphers

1. First proposal [KM10] — 3-round Feistel distinguisher
2. Even-Mansour key-recovery [KM12]:



The EM cipher

[KM12]

$$\mathsf{Enc}_{k_1,k_2}(m) = P(m \oplus k_1) \oplus k_2$$

$$f_{k_1}(m) = \mathsf{Enc}_{k_1,k_2}(m) \oplus P(m)$$

$$f_{k_1}(m) = P(m \oplus k_1) \oplus k_2 \oplus P(m)$$

$$f_{k_1}(m) = f(m \oplus k_1)$$

Simon's algorithm to recover k_1

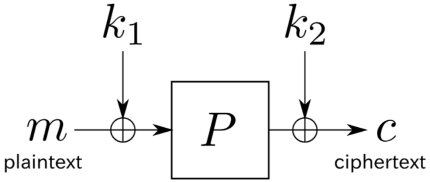
$$k_2 = \mathsf{Enc}_{k_1,k_2}(m) \oplus P(m \oplus k_1)$$

Superposition Attacks

Block Ciphers

1. First proposal [KM10] — 3-round Feistel distinguisher

2. Even-Mansour key-recovery [KM12]:



The EM cipher

[KM12]

$$\mathsf{Enc}_{k_1,k_2}(m) = P(m \oplus k_1) \oplus k_2$$

$$f_{k_1}(m) = \mathsf{Enc}_{k_1,k_2}(m) \oplus P(m)$$

$$f_{k_1}(m) = P(m \oplus k_1) \oplus k_2 \oplus P(m)$$

$$f_{k_1}(m) = f(m \oplus k_1)$$

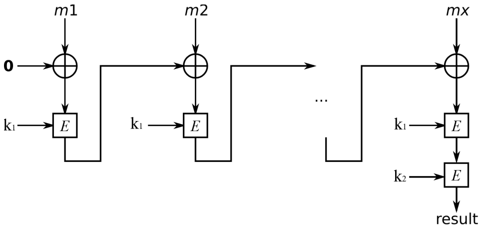
Simon's algorithm to recover k_1

$$k_2 = \mathsf{Enc}_{k_1,k_2}(m) \oplus P(m \oplus k_1)$$

Superposition Attacks

MACs

- Forgery attack — CBC-MAC [KLLN16]
- *More:* LightMAC, PolyMAC, GCM-SIV2, Poly1305, ... [BLNS21]



Source: wikipedia

$$\text{CBCMAC}(m_1|m_2) = E_{k_2}\left(E_{k_1}\left(m_2 \oplus E_{k_1}\left(m_1\right)\right)\right)$$

$$f : \mathbb{Z}_2 \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$$
$$b, x \to \text{CBCMAC}(m_b|x)$$

$$f(b,x) = E_{k_2}(E_{k_1}(x \oplus E_{k_1}(m_b)))$$

$$f(b,x) = f(b \oplus 1, x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1))$$

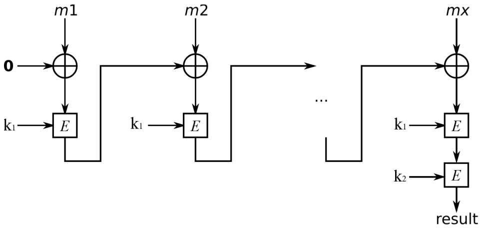
Simon's: $\Rightarrow E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

1. Pick x
2. Query $m_0|x$
3. Return $m_1|x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

Superposition Attacks

MACs

- Forgery attack — CBC-MAC [KLLN16]
- *More:* LightMAC, PolyMAC, GCM-SIV2, Poly1305, . . . [BLNS21]



Source: wikipedia

$$\text{CBCMAC}(m_1|m_2) = E_{k_2}\left(E_{k_1}\left(m_2 \oplus E_{k_1}\left(m_1\right)\right)\right)$$

$$f : \mathbb{Z}_2 \times \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^n$$
$$b, x \rightarrow \text{CBCMAC}(m_b|x)$$

$$f(b,x) = E_{k_2}(E_{k_1}(x \oplus E_{k_1}(m_b)))$$

$$f(b,x) = f(b \oplus 1, x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1))$$

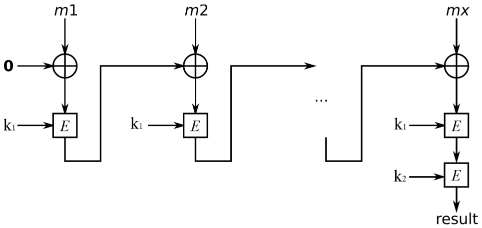
Simon's: $\Rightarrow E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

1. Pick x
2. Query $m_0|x$
3. Return $m_1|x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

Superposition Attacks

MACs

- Forgery attack — CBC-MAC [KLLN16]
- *More:* LightMAC, PolyMAC, GCM-SIV2, Poly1305, ... [BLNS21]



Source: wikipedia

$$\text{CBCMAC}(m_1|m_2) = E_{k_2}\left(E_{k_1}\left(m_2 \oplus E_{k_1}\left(m_1\right)\right)\right)$$

$$f : \mathbb{Z}_2 \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$$
$$b, x \to \text{CBCMAC}(m_b|x)$$

$$f(b,x) = E_{k_2}(E_{k_1}(x \oplus E_{k_1}(m_b)))$$

$$f(b,x) = f(b \oplus 1, x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1))$$

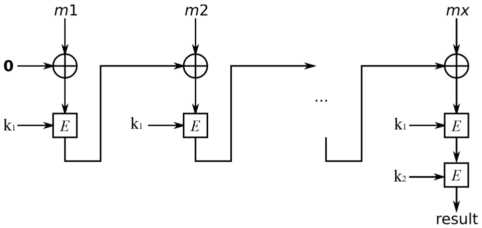
Simon's: $\Rightarrow E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

1. Pick x
2. Query $m_0|x$
3. Return $m_1|x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

Superposition Attacks

MACs

- Forgery attack — CBC-MAC [KLLN16]
- *More:* LightMAC, PolyMAC, GCM-SIV2, Poly1305, . . . [BLNS21]



Source: wikipedia

$\text{CBCMAC}(m_1|m_2) =$

$E_{k_2}\left(E_{k_1}\left(m_2 \oplus E_{k_1}\left(m_1\right)\right)\right)$

$$f : \mathbb{Z}_2 \times \mathbb{Z}_2^n \to \mathbb{Z}_2^n$$
$$b, x \to \text{CBCMAC}(m_b|x)$$

$$f(b,x) = E_{k_2}(E_{k_1}(x \oplus E_{k_1}(m_b)))$$

$$f(b,x) = f(b \oplus 1, x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1))$$

Simon's: $\Rightarrow E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

1. Pick x
2. Query $m_0|x$
3. Return $m_1|x \oplus E_{k_1}(m_0) \oplus E_{k_1}(m_1)$

Contents

Discussion

- New quantum algorithms with applications to cryptanalysis (both Q1 and Q2 models).
- How will cryptographic primitives be implemented in **quantum networks**?
- Are the new **post-quantum** security assumptions quantum-secure?
- Are the new post-quantum proposals also **Q2-secure**?
- How to model oracles in the presence of a quantum computer?

What are the appropriate **security models**?

Discussion

- New quantum algorithms with applications to cryptanalysis (both Q1 and Q2 models).
- How will cryptographic primitives be implemented in **quantum networks**?
- Are the new **post-quantum** security assumptions quantum-secure?
- Are the new post-quantum proposals also **Q2-secure**?
- How to model oracles in the presence of a quantum computer?

What are the appropriate **security models**?

Discussion

- New quantum algorithms with applications to cryptanalysis (both Q1 and Q2 models).
- How will cryptographic primitives be implemented in **quantum networks**?
- Are the new **post-quantum** security assumptions quantum-secure?
- Are the new post-quantum proposals also **Q2-secure**?
- How to model oracles in the presence of a quantum computer?

What are the appropriate **security models**?

Discussion

- New quantum algorithms with applications to cryptanalysis (both Q1 and Q2 models).
- How will cryptographic primitives be implemented in **quantum networks**?
- Are the new **post-quantum** security assumptions quantum-secure?
- Are the new post-quantum proposals also **Q2-secure**?
- How to model oracles in the presence of a quantum computer?

What are the appropriate **security models**?

Discussion

- New quantum algorithms with applications to cryptanalysis (both Q1 and Q2 models).
- How will cryptographic primitives be implemented in **quantum networks**?
- Are the new **post-quantum** security assumptions quantum-secure?
- Are the new post-quantum proposals also **Q2-secure**?
- How to model oracles in the presence of a quantum computer?

What are the appropriate **security models**?

Discussion

- New quantum algorithms with applications to cryptanalysis (both Q1 and Q2 models).
- How will cryptographic primitives be implemented in **quantum networks**?
- Are the new **post-quantum** security assumptions quantum-secure?
- Are the new post-quantum proposals also **Q2-secure**?
- How to model oracles in the presence of a quantum computer?

What are the appropriate **security models**?

Discussion

- New quantum algorithms with applications to cryptanalysis (both Q1 and Q2 models).
- How will cryptographic primitives be implemented in **quantum networks**?
- Are the new **post-quantum** security assumptions quantum-secure?
- Are the new post-quantum proposals also **Q2-secure**?
- How to model oracles in the presence of a quantum computer?

What are the appropriate **security models**?