A Meta-Complexity Characterisation of Quantum Cryptography

Bruno P. Cavalar

University of Oxford

Joint work with
Eli Goldin (NYU), Matthew Gray (Oxford), Peter Hall (NYU)



EUROCRYPT 2025

The holy grail of cryptography?

Minimal complexity-theoretic assumption required for cryptography?

- $P \neq NP$: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if $BQP = QMA$ (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

The holy grail of cryptography?

Minimal complexity-theoretic assumption required for cryptography?

- P $\neq$ NP: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if BQP $=$ QMA (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

The holy grail of cryptography?

Minimal complexity-theoretic assumption required for cryptography?

- P $\neq$ NP: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if BQP $=$ QMA (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

Minimal complexity-theoretic assumption required for cryptography?

- $P \neq NP$: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if $BQP = QMA$ (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

Minimal complexity-theoretic assumption required for cryptography?

- P $\neq$ NP: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if BQP $=$ QMA (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

The holy grail of cryptography?

Minimal complexity-theoretic assumption required for cryptography?

- P $\neq$ NP: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if BQP = QMA (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

Minimal complexity-theoretic assumption required for cryptography?

- $\mathsf{P} \neq \mathsf{NP}$: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if $\mathsf{BQP} = \mathsf{QMA}$ (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

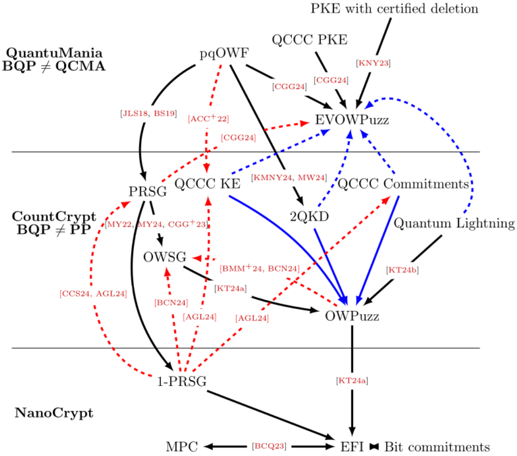
This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

Minimal complexity-theoretic assumption required for cryptography?

- $P \neq NP$: necessary; sufficient?
  - Specific constructions: Factoring, LWE, etc.
- Quantum complication: PRSs may exist even if $BQP = QMA$ (in particular, $\nexists$ pqOWFs) [Kretschmer, TQC 2021]
  - The classical "holy grail" is no longer relevant
- Meta-complexity: classical characterisations!

This talk: an equivalence between **quantum** cryptography and complexity theory *via meta-complexity*.

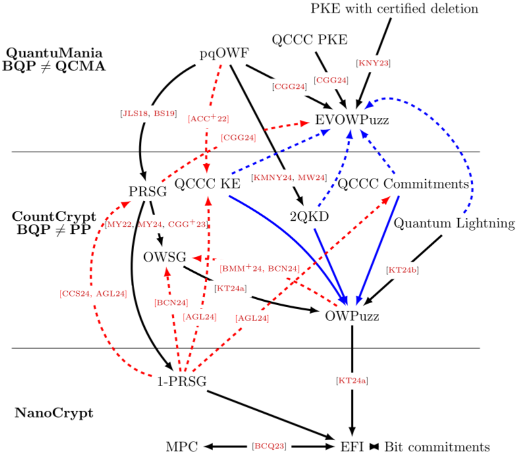
Minimal quantum cryptographic primitive?



Credits: Goldin, Morimae, Mutreja, and Yamakawa [2024]

- Primitives following from PRS are not known to be equivalent (in fact, plenty of oracle separations)
- OWPuzz's seem minimal in QCCC and CountCrypt
- EFI seems minimal for quantum communication
- **This work:** OWPuzz

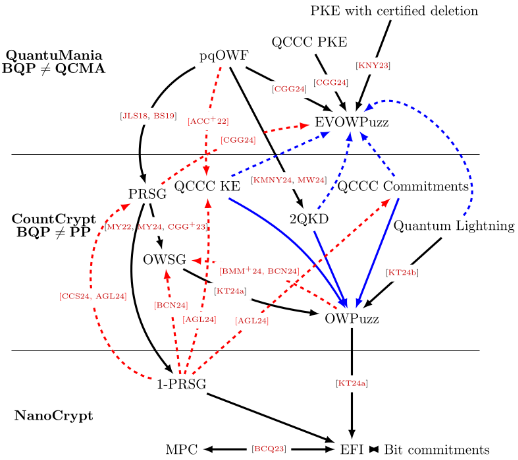
Minimal quantum cryptographic primitive?



Credits: Goldin, Morimae, Mutreja, and Yamakawa [2024]

- Primitives following from PRS are not known to be equivalent (in fact, plenty of oracle separations)
- OWPuzz's seem minimal in QCCC and CountCrypt
- EFI seems minimal for quantum communication
- **This work:** OWPuzz

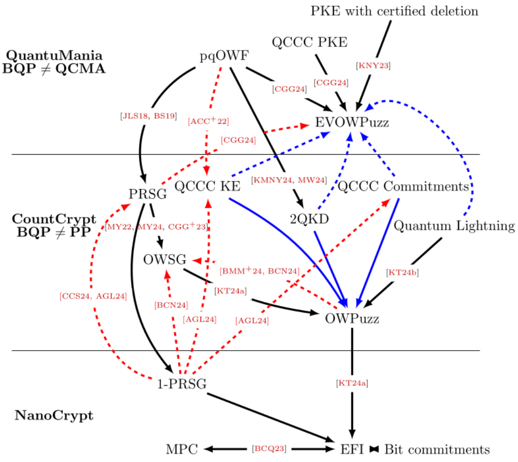
Minimal quantum cryptographic primitive?



Credits: Goldin, Morimae, Mutreja, and Yamakawa [2024]

- Primitives following from PRS are not known to be equivalent (in fact, plenty of oracle separations)
- OWPuzz's seem minimal in QCCC and CountCrypt
- EFI seems minimal for quantum communication
- **This work:** OWPuzz

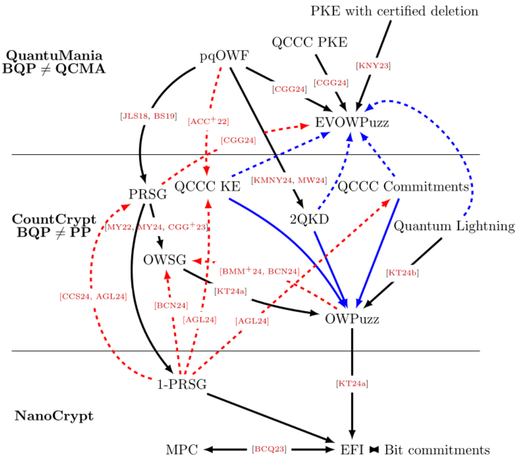
Minimal quantum cryptographic primitive?



Credits: Goldin, Morimae, Mutreja, and Yamakawa [2024]

- Primitives following from PRS are not known to be equivalent (in fact, plenty of oracle separations)
- OWPuzz's seem minimal in QCCC and CountCrypt
- EFI seems minimal for quantum communication
- **This work:** OWPuzz

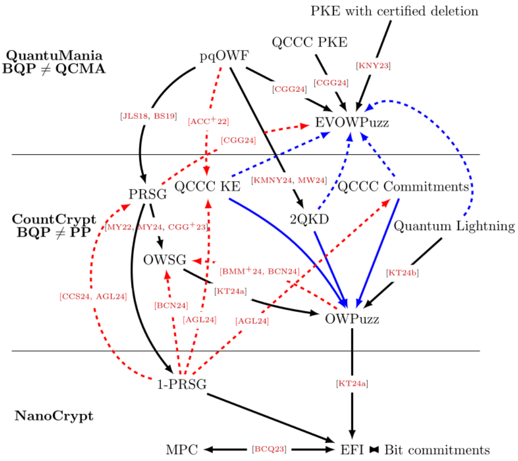
Minimal quantum cryptographic primitive?



Credits: Goldin, Morimae, Mutreja, and Yamakawa [2024]

- Primitives following from PRS are not known to be equivalent (in fact, plenty of oracle separations)
- OWPuzz's seem minimal in QCCC and CountCrypt
- EFI seems minimal for quantum communication
- **This work:** OWPuzz

Minimal quantum cryptographic primitive?



Credits: Goldin, Morimae, Mutreja, and Yamakawa [2024]

- Primitives following from PRS are not known to be equivalent (in fact, plenty of oracle separations)
- OWPuzz's seem minimal in QCCC and CountCrypt
- EFI seems minimal for quantum communication
- **This work:** OWPuzz

One-way puzzles

- OWPuzz = (Samp, Ver). [Khurana-Tomer, STOC 2024]
- $\mathsf{Samp}(1^\lambda) \to (k, s) \in \{0,1\}^*$ in quantum polynomial-time (QPT).
  - *k : key*; *s: puzzle*.
- *Soundness:* $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(k,s) = \top] = 1 - \mathsf{negl}(n)$
- *Correctness:* $\forall$QPT$\mathcal{A}$, $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(\mathcal{A}(s), s) = \top] = \mathsf{negl}(n)$
- Notably: Ver can be inefficient!
- Arises naturally from *shadow tomography*.

One-way puzzles

- OWPuzz = (Samp, Ver). [Khurana-Tomer, STOC 2024]
- $\mathsf{Samp}(1^\lambda) \to (k, s) \in \{0,1\}^*$ in quantum polynomial-time (QPT).
  - k : *key*; s: *puzzle*.
- *Soundness:* $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(k,s) = \top] = 1 - \mathsf{negl}(n)$
- *Correctness:* $\forall$QPT$\mathcal{A}$, $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(\mathcal{A}(s), s) = \top] = \mathsf{negl}(n)$
- Notably: Ver can be inefficient!
- Arises naturally from *shadow tomography*.

One-way puzzles

- OWPuzz = (Samp, Ver). [Khurana-Tomer, STOC 2024]
- $\mathsf{Samp}(1^\lambda) \to (k, s) \in \{0,1\}^*$ in quantum polynomial-time (QPT).
  - k : *key*; s: *puzzle*.
- *Soundness:* $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(k,s) = \top] = 1 - \mathsf{negl}(n)$
- *Correctness:* $\forall$QPT$\mathcal{A}$, $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(\mathcal{A}(s), s) = \top] = \mathsf{negl}(n)$
- Notably: Ver can be inefficient!
- Arises naturally from *shadow tomography*.

One-way puzzles

- OWPuzz = (Samp, Ver). [Khurana-Tomer, STOC 2024]
- $\mathsf{Samp}(1^\lambda) \to (k, s) \in \{0,1\}^*$ in quantum polynomial-time (QPT).
  - *k : key*; *s: puzzle*.
- *Soundness:* $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(k,s) = \top] = 1 - \mathsf{negl}(n)$
- *Correctness:* $\forall \mathsf{QPT}\,\mathcal{A}$, $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(\mathcal{A}(s), s) = \top] = \mathsf{negl}(n)$
- Notably: Ver can be inefficient!
- Arises naturally from *shadow tomography*.

One-way puzzles

- OWPuzz = (Samp, Ver). [Khurana-Tomer, STOC 2024]
- $\mathsf{Samp}(1^\lambda) \to (k, s) \in \{0,1\}^*$ in quantum polynomial-time (QPT).
  - k : *key*; s: *puzzle*.
- *Soundness:* $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(k,s) = \top] = 1 - \mathsf{negl}(n)$
- *Correctness:* $\forall$QPT$\mathcal{A}$, $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(\mathcal{A}(s), s) = \top] = \mathsf{negl}(n)$
- Notably: Ver can be inefficient!
- Arises naturally from *shadow tomography*.

One-way puzzles

- OWPuzz = (Samp, Ver). [Khurana-Tomer, STOC 2024]
- $\mathsf{Samp}(1^\lambda) \to (k, s) \in \{0,1\}^*$ in quantum polynomial-time (QPT).
  - k : *key*; s: *puzzle*.
- *Soundness:* $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(k,s) = \top] = 1 - \mathsf{negl}(n)$
- *Correctness:* $\forall$QPT$\mathcal{A}$, $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(\mathcal{A}(s), s) = \top] = \mathsf{negl}(n)$
- Notably: Ver can be inefficient!
- Arises naturally from *shadow tomography*.

One-way puzzles

- OWPuzz = (Samp, Ver). [Khurana-Tomer, STOC 2024]
- $\mathsf{Samp}(1^\lambda) \to (k, s) \in \{0,1\}^*$ in quantum polynomial-time (QPT).
  - k : *key*; s: *puzzle*.
- *Soundness:* $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(k,s) = \top] = 1 - \mathsf{negl}(n)$
- *Correctness:* $\forall \mathrm{QPT}\,\mathcal{A}$, $\mathbb{P}_{(k,s)\leftarrow \mathsf{Samp}(1^\lambda)}[\mathsf{Ver}(\mathcal{A}(s), s) = \top] = \mathsf{negl}(n)$
- Notably: Ver can be inefficient!
- Arises naturally from *shadow tomography*.

Kolmogorov Complexity

How much information is in a text?

- Example 1: “1010101010101010” (Print “10" 8 times)
- Example 2: “0001010100001011” (???)

Kolmogorov complexity $\mathrm{K}(x)$ of a string $x \in \{0,1\}^*$:
minimum length of a program that outputs x.

PROPERTIES

1. $\mathrm{K}(x) \leqslant |x|$.
2. Random strings have near maximum Kolmogorov complexity.
3. Computing $\mathrm{K}(x)$ is impossible!

Kolmogorov Complexity

How much information is in a text?

- Example 1: “1010101010101010” (Print “10" 8 times)
- Example 2: “0001010100001011” (???)

Kolmogorov complexity $\mathrm{K}(x)$ of a string $x \in \{0,1\}^*$:
minimum length of a program that outputs x.

PROPERTIES

1. $\mathrm{K}(x) \leqslant |x|$.
2. Random strings have near maximum Kolmogorov complexity.
3. Computing $\mathrm{K}(x)$ is impossible!

Kolmogorov Complexity

How much information is in a text?

- Example 1: “1010101010101010” (Print “10" 8 times)
- Example 2: “0001010100001011” (???)

Kolmogorov complexity $\mathrm{K}(x)$ of a string $x \in \{0,1\}^*$:
minimum length of a program that outputs x.

PROPERTIES

1. $\mathrm{K}(x) \leqslant |x|$.
2. Random strings have near maximum Kolmogorov complexity.
3. Computing $\mathrm{K}(x)$ is impossible!

Kolmogorov Complexity

How much information is in a text?

- Example 1: “1010101010101010” (Print “10" 8 times)
- Example 2: “0001010100001011” (???)

Kolmogorov complexity $\mathrm{K}(x)$ of a string $x \in \{0,1\}^*$:
minimum length of a program that outputs x.

PROPERTIES

1. $\mathrm{K}(x) \leqslant |x|$.
2. Random strings have near maximum Kolmogorov complexity.
3. Computing $\mathrm{K}(x)$ is impossible!

Kolmogorov Complexity

How much information is in a text?

- Example 1: “1010101010101010” (Print “10" 8 times)
- Example 2: “0001010100001011” (???)

Kolmogorov complexity $\mathrm{K}(x)$ of a string $x \in \{0,1\}^*$:
minimum length of a program that outputs x.

PROPERTIES

1. $\mathrm{K}(x) \leqslant |x|$.
2. Random strings have near maximum Kolmogorov complexity.
3. Computing $\mathrm{K}(x)$ is impossible!

Kolmogorov Complexity

How much information is in a text?

- Example 1: “1010101010101010” (Print “10" 8 times)
- Example 2: “0001010100001011” (???)

Kolmogorov complexity $\mathsf{K}(x)$ of a string $x \in \{0,1\}^*$:
minimum length of a program that outputs x.

PROPERTIES

1. $\mathsf{K}(x) \leqslant |x|$.
2. Random strings have near maximum Kolmogorov complexity.
3. Computing $\mathsf{K}(x)$ is impossible!

Kolmogorov Complexity

How much information is in a text?

- Example 1: “1010101010101010” (Print “10" 8 times)
- Example 2: “0001010100001011” (???)

Kolmogorov complexity $\mathrm{K}(x)$ of a string $x \in \{0,1\}^*$:
minimum length of a program that outputs x.

PROPERTIES

1. $\mathrm{K}(x) \leqslant |x|$.
2. Random strings have near maximum Kolmogorov complexity.
3. Computing $\mathrm{K}(x)$ is impossible!

Approximating Kolmogorov Complexity (GapK)

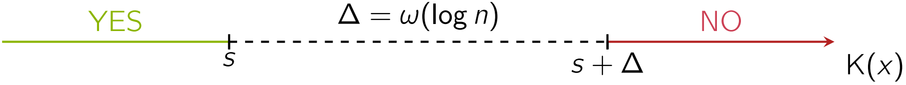
Let GapK$[s, s+\Delta]$ be the "promise" problem of distinguishing between strings of K.c. at most s and those with K.c. at least $s+\Delta$.

IRS'20: GapK can be solved in exponential-time *on average* (PP = PostBQP).

For every PPT-samplable distribution $\mathcal{D}$ and $\Delta = \omega(\log n)$, it's possible to compute GapK$[s, s+\Delta]$ on $\mathcal{D}$ with error at most $n^{-O(1)}$ in exponential time.
[IRS, STOC20]

Approximating Kolmogorov Complexity (GapK)

Let GapK$[s, s+\Delta]$ be the "promise" problem of distinguishing between strings of K.c. at most s and those with K.c. at least $s+\Delta$.

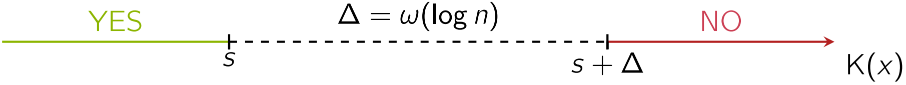


IRS'20: GapK can be solved in exponential-time *on average* (PP = PostBQP).

For every PPT-samplable distribution $\mathcal{D}$ and $\Delta = \omega(\log n)$, it's possible to compute GapK$[s, s+\Delta]$ on $\mathcal{D}$ with error at most $n^{-O(1)}$ in exponential time.
[IRS, STOC20]

Approximating Kolmogorov Complexity (GapK)

Let GapK$[s, s+\Delta]$ be the "promise" problem of distinguishing between strings of K.c. at most s and those with K.c. at least $s+\Delta$.

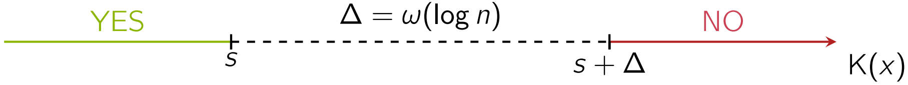


IRS'20: GapK can be solved in exponential-time *on average* (PP = PostBQP).

For every PPT-samplable distribution $\mathcal{D}$ and $\Delta = \omega(\log n)$, it's possible to compute GapK$[s, s+\Delta]$ on $\mathcal{D}$ with error at most $n^{-O(1)}$ in exponential time.
[IRS, STOC20]

Approximating Kolmogorov Complexity (GapK)

Let GapK$[s, s+\Delta]$ be the "promise" problem of distinguishing between strings of K.c. at most s and those with K.c. at least $s+\Delta$.



IRS'20: GapK can be solved in exponential-time *on average* (PP = PostBQP).

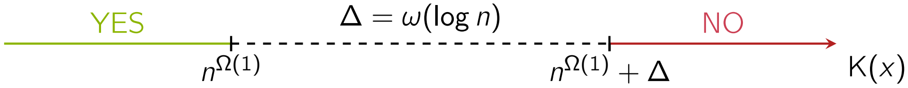
For every PPT-samplable distribution $\mathcal{D}$ and $\Delta = \omega(\log n)$, it's possible to compute GapK$[s, s+\Delta]$ on $\mathcal{D}$ with error at most $n^{-O(1)}$ in exponential time.
[IRS, STOC20]

Cryptography vs. Kolmogorov complexity

Theorem (Ilango-Ren-Santhanam [IRS], STOC 2020)

The following are equivalent:

- *One-way functions exist.*
- *For some* $s = n^{\Omega(1)}$ *and* $\Delta = \omega(\log n)$, *there exists a samplable distribution* $\mathcal{D}$ *such that* $\mathrm{GapK}[s, s+\Delta]$ *is average-case hard on* $\mathcal{D}$.



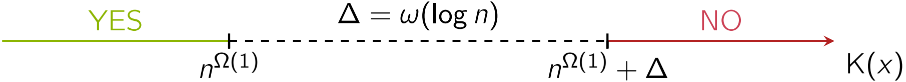
The OWF encodes a hard probability distribution, and vice-versa.

Quantum Cryptography vs. Kolmogorov complexity

Theorem (C.-Goldin-Gray-Hall [CGGH], EUROCRYPT 2025)

The following are equivalent:

- *One-way* **puzzles** *exist.*
- *For some* $s = n^{\Omega(1)}$ *and* $\Delta = \omega(\log n)$, *there exists a* **quantum** *samplable distribution* $\mathcal{D}$ *such that* $\mathrm{GapK}[s, s+\Delta]$ *is average-case hard on* $\mathcal{D}$ **for quantum algorithms**.



The OWPuzz encodes a hard probability distribution, and vice-versa.

Quantum Cryptography vs. Kolmogorov complexity

Theorem (C.-Goldin-Gray-Hall [CGGH], EUROCRYPT 2025)

The following are equivalent:

- *One-way* **puzzles** *exist.*
- *For some* $s = n^{\Omega(1)}$ *and* $\Delta = \omega(\log n)$*, there exists a* **quantum** *samplable distribution* $\mathcal{D}$ *such that* $\mathrm{GapK}[s, s + \Delta]$ *is average-case hard on* $\mathcal{D}$ **for quantum algorithms**.

- Further evidence of OWPuzz's centrality in QCCC!
  - Besides combiners, hardness amplification, etc.
  - Natural generalisation of OWFs
  - Embodies some fundamental hardness via meta-complexity

Probability estimation: crucial intermediate step

Let $\mathcal{D}$ be a sampler. Let $p_x := \mathbb{P}[x \text{ is sampled by } \mathcal{D}]$.
δ-Prob. estimation: given $x \leftarrow \mathcal{D}$, output $\mathcal{A}(x) \in [\delta \cdot p_x, p_x]$ w.h.p..

$$\text{Breaking OWPuzz} \implies \text{Prob. estimation}$$

Classically: use the random bits of the sampler to construct a OWF candidate:

$$f_k(h, r) = \mathcal{D}(r), h, h(r),$$

where h is a hash function mapping to k bits.
Estimate p_x by sampling h, z and inverting (x, h, z) (many times).
(Intuition: select a random hash bucket and see if r "is there")

Quantumly: randomness is inherent to the hard distribution!

Probability estimation: crucial intermediate step

Let $\mathcal{D}$ be a sampler. Let $p_x := \mathbb{P}[x \text{ is sampled by } \mathcal{D}]$.

δ-Prob. estimation: given $x \leftarrow \mathcal{D}$, output $\mathcal{A}(x) \in [\delta \cdot p_x, p_x]$ w.h.p..

$$\text{Breaking OWPuzz} \implies \text{Prob. estimation}$$

Classically: use the random bits of the sampler to construct a OWF candidate:

$$f_k(h, r) = \mathcal{D}(r), h, h(r),$$

where h is a hash function mapping to k bits.
Estimate p_x by sampling h, z and inverting (x, h, z) (many times).
(Intuition: select a random hash bucket and see if r "is there")

Quantumly: randomness is inherent to the hard distribution!

Probability estimation: crucial intermediate step

Let $\mathcal{D}$ be a sampler. Let $p_x := \mathbb{P}[x \text{ is sampled by } \mathcal{D}]$.
δ-Prob. estimation: given $x \leftarrow \mathcal{D}$, output $\mathcal{A}(x) \in [\delta \cdot p_x, p_x]$ w.h.p..

Breaking OWPuzz $\Longrightarrow$ Prob. estimation

Classically: use the random bits of the sampler to construct a OWF candidate:

$$f_k(h, r) = \mathcal{D}(r), h, h(r),$$

where h is a hash function mapping to k bits.
Estimate p_x by sampling h, z and inverting (x, h, z) (many times).
(Intuition: select a random hash bucket and see if r "is there")

Quantumly: randomness is inherent to the hard distribution!

Probability estimation: crucial intermediate step

Let $\mathcal{D}$ be a sampler. Let $p_x := \mathbb{P}[x \text{ is sampled by } \mathcal{D}]$.
δ-Prob. estimation: given $x \leftarrow \mathcal{D}$, output $\mathcal{A}(x) \in [\delta \cdot p_x, p_x]$ w.h.p..

$$\text{Breaking OWPuzz} \implies \text{Prob. estimation}$$

Classically: use the random bits of the sampler to construct a OWF candidate:

$$f_k(h, r) = \mathcal{D}(r), h, h(r),$$

where h is a hash function mapping to k bits.
Estimate p_x by sampling h, z and inverting (x, h, z) (many times).
(Intuition: select a random hash bucket and see if r "is there")

Quantumly: randomness is inherent to the hard distribution!

Probability estimation: crucial intermediate step

Let $\mathcal{D}$ be a sampler. Let $p_x := \mathbb{P}[x \text{ is sampled by } \mathcal{D}]$.
δ-Prob. estimation: given $x \leftarrow \mathcal{D}$, output $\mathcal{A}(x) \in [\delta \cdot p_x, p_x]$ w.h.p..

$$\text{Breaking OWPuzz} \implies \text{Prob. estimation}$$

Classically: use the random bits of the sampler to construct a OWF candidate:

$$f_k(h, r) = \mathcal{D}(r), h, h(r),$$

where h is a hash function mapping to k bits.
Estimate p_x by sampling h, z and inverting (x, h, z) (many times).
(Intuition: select a random hash bucket and see if r "is there")

Quantumly: randomness is inherent to the hard distribution!

Probability estimation: crucial intermediate step

Let $\mathcal{D}$ be a sampler. Let $p_x := \mathbb{P}[x \text{ is sampled by } \mathcal{D}]$.
δ-Prob. estimation: given $x \leftarrow \mathcal{D}$, output $\mathcal{A}(x) \in [\delta \cdot p_x, p_x]$ w.h.p..

$$\text{Breaking OWPuzz} \implies \text{Prob. estimation}$$

Classically: use the random bits of the sampler to construct a OWF candidate:

$$f_k(h, r) = \mathcal{D}(r), h, h(r),$$

where h is a hash function mapping to k bits.
Estimate p_x by sampling h, z and inverting (x, h, z) (many times).
(Intuition: select a random hash bucket and see if r "is there")

Quantumly: randomness is inherent to the hard distribution!

Probability estimation: crucial intermediate step

Let $\mathcal{D}$ be a sampler. Let $p_x := \mathbb{P}[x \text{ is sampled by } \mathcal{D}]$.
δ-Prob. estimation: given $x \leftarrow \mathcal{D}$, output $\mathcal{A}(x) \in [\delta \cdot p_x, p_x]$ w.h.p..

$$\text{Breaking OWPuzz} \Longrightarrow \text{Prob. estimation}$$

Classically: use the random bits of the sampler to construct a OWF candidate:

$$f_k(h, r) = \mathcal{D}(r), h, h(r),$$

where h is a hash function mapping to k bits.
Estimate p_x by sampling h, z and inverting (x, h, z) (many times).
(Intuition: select a random hash bucket and see if r "is there")

Quantumly: randomness is inherent to the hard distribution!

Distributional one-way puzzles

On Central Primitives for Quantum Cryptography with Classical Communication

Kai-Min Chung[1], Eli Goldin[2], and Matthew Gray[3]

[1]Academia Sinica (kmchung@iis.sinica.edu.tw)
[2]New York University (eli.goldin@nyu.edu)
[3]University of Oxford (matthew.gray@cs.ox.ac.uk)

- One-way puzzles $\iff$ Distributional one-way puzzles
- **Takeaway:** $\nexists$ OWPuzz $\implies$ sample from (k, s) *conditioned* on s.

Distributional one-way puzzles

On Central Primitives for Quantum Cryptography with Classical Communication

Kai-Min Chung[1], Eli Goldin[2], and Matthew Gray[3]

[1]Academia Sinica (kmchung@iis.sinica.edu.tw)
[2]New York University (eli.goldin@nyu.edu)
[3]University of Oxford (matthew.gray@cs.ox.ac.uk)



- One-way puzzles $\Longleftrightarrow$ Distributional one-way puzzles
- **Takeaway:** $\nexists$ OWPuzz $\Longrightarrow$ sample from (k, s) *conditioned* on s.

Amplitude estimation with a distributional inverter

One-way puzzle: sample $x \leftarrow \mathcal{D}$, h hash function mapping to k bits.
Key: x, Puzzle: $(h, h(x))$.
Distributionally invert [CGG]: as k increases, x is isolated by $(h, h(x))$.
(Intuition: fix the "right" random hash bucket, and see if anything else besides x is there.)
The threshold when x starts to become isolated is our estimate for p_x.

Amplitude estimation with a distributional inverter

One-way puzzle: sample $x \leftarrow \mathcal{D}$, h hash function mapping to k bits.
Key: x, Puzzle: $(h, h(x))$.
Distributionally invert [CGG]: as k increases, x is isolated by $(h, h(x))$.
(Intuition: fix the “right” random hash bucket, and see if anything else besides x is there.)
The threshold when x starts to become isolated is our estimate for p_x.

Amplitude estimation with a distributional inverter

One-way puzzle: sample $x \leftarrow \mathcal{D}$, h hash function mapping to k bits.
Key: x, Puzzle: $(h, h(x))$.
Distributionally invert [CGG]: as k increases, x is isolated by $(h, h(x))$.
(Intuition: fix the "right" random hash bucket, and see if anything else besides x is there.)
The threshold when x starts to become isolated is our estimate for p_x.

Computing GapK with probability estimation

Just like the classical case [IRS].
Requires a "quantum coding theorem":

$$\mathrm{K}(x) \lessapprox \log\left(\frac{1}{\mathbb{P}[\mathcal{D} \to x]}\right) + |\mathcal{D}|\,.$$

We then use probability as a proxy for Kolmogorov complexity.
(In other words, low probability $\approx$ high Kolmogorov complexity.)

Computing GapK with probability estimation

Just like the classical case [IRS].
Requires a "quantum coding theorem":

$$\mathsf{K}(x) \lessapprox \log\left(\frac{1}{\mathbb{P}[\mathcal{D} \to x]}\right) + |\mathcal{D}|\,.$$

We then use probability as a proxy for Kolmogorov complexity.
(In other words, low probability $\approx$ high Kolmogorov complexity.)

Breaking one-way puzzles with GapK

CGG: OWPuzz $\implies$ "non-uniform entropy-gap generator".



A **QPT** distribution $\mathcal{D}$ whose entropy is sufficiently far from maximum and which is **QPT**-indistinguishable from uniform.

By coding theorem, low entropy $\approx$ low Kolmogorov complexity.
Thus, GapK can break the entropy-gap generator!

Breaking one-way puzzles with GapK

CGG: OWPuzz $\Longrightarrow$ "non-uniform entropy-gap generator".

A **QPT** distribution $\mathcal{D}$ whose entropy is sufficiently far from maximum and which is **QPT**-indistinguishable from uniform.

By coding theorem, low entropy $\approx$ low Kolmogorov complexity.
Thus, GapK can break the entropy-gap generator!

Concurrent work

1. Khurana-Tomer (STOC 2025): Shows equivalence between probability estimation and one-way puzzles (with tighter approximation factor);
2. Hiroka-Morimae (2024): Shows how to construct one-way puzzles from average-case hardness of GapK (with a different proof).

Open questions

1. Characterising with a problem with worst-case complexity bound? (Classical: AM).
2. Characterise other primitives? E.g., EFIs, OWSGs.
3. Strong average-case hardness?
4. Oracle separation does not rule out: construct OWPuzzles from the hardness of a (meta-complexity?) problem in QMA.

Open questions

1. Characterising with a problem with worst-case complexity bound? (Classical: AM).
2. Characterise other primitives? E.g., EFIs, OWSGs.
3. Strong average-case hardness?
4. Oracle separation does not rule out: construct OWPuzzles from the hardness of a (meta-complexity?) problem in QMA.

Open questions

1. Characterising with a problem with worst-case complexity bound? (Classical: AM).
2. Characterise other primitives? E.g., EFIs, OWSGs.
3. Strong average-case hardness?
4. Oracle separation does not rule out: construct OWPuzzles from the hardness of a (meta-complexity?) problem in QMA.

Open questions

1. Characterising with a problem with worst-case complexity bound? (Classical: AM).
2. Characterise other primitives? E.g., EFIs, OWSGs.
3. Strong average-case hardness?
4. Oracle separation does not rule out: construct OWPuzzles from the hardness of a (meta-complexity?) problem in QMA.

Open questions

1. Characterising with a problem with worst-case complexity bound? (Classical: AM).
2. Characterise other primitives? E.g., EFIs, OWSGs.
3. Strong average-case hardness?
4. Oracle separation does not rule out: construct OWPuzzles from the hardness of a (meta-complexity?) problem in QMA.

Thanks!