Proofs of Quantumness

Alexandru Gheorghiu (ETH Zürich → Chalmers University of Technology)

Article

Quantum supremacy using a programmable superconducting processor

https://doi.org/10.1038/s41586-019-1666-5

Received: 22 July 2019

Accepted: 20 September 2019

Published online: 23 October 2019

Frank Arute[1], Kunal Arya[1], Ryan Babbush[1], Dave Bacon[1], Joseph C. Bardin[1,2], Rami Barends[1], Rupak Biswas[3], Sergio Boixo[1], Fernando G. S. L. Brandao[1,4], David A. Buell[1], Brian Burkett[1], Yu Chen[1], Zijun Chen[1], Ben Chiaro[5], Roberto Collins[1], William Courtney[1], Andrew Dunsworth[1], Edward Farhi[1], Brooks Foxen[1,5], Austin Fowler[1], Craig Gidney[1], Marissa Giustina[1], Rob Graff[1], Keith Guerin[1], Steve Habegger[1], Matthew P. Harrigan[1], Michael J. Hartmann[1,6], Alan Ho[1], Markus Hoffmann[1], Trent Huang[1], Travis S. Humble[7], Sergei V. Isakov[1], Evan Jeffrey[1], Zhang Jiang[1], Dvir Kafri[1], Kostyantyn Kechedzhi[1], Julian Kelly[1], Paul V. Klimov[1], Sergey Knysh[1], Alexander Korotkov[1,8], Fedor Kostritsa[1], David Landhuis[1], Mike Lindmark[1], Erik Lucero[1], Dmitry Lyakh[9], Salvatore Mandrà[3,10], Jarrod R. McClean[1], Matthew McEwen[5], Anthony Megrant[1], Xiao Mi[1], Kristel Michielsen[11,12], Masoud Mohseni[1], Josh Mutus[1], Ofer Naaman[1], Matthew Neeley[1], Charles Neill[1], Murphy Yuezhen Niu[1], Eric Ostby[1], Andre Petukhov[1], John C. Platt[1], Chris Quintana[1], Eleanor G. Rieffel[3], Pedram Roushan[1], Nicholas C. Rubin[1], Daniel Sank[1], Kevin J. Satzinger[1], Vadim Smelyanskiy[1], Kevin J. Sung[1,13], Matthew D. Trevithick[1], Amit Vainsencher[1], Benjamin Villalonga[1,14], Theodore White[1], Z. Jamie Yao[1], Ping Yeh[1], Adam Zalcman[1], Hartmut Neven[1] & John M. Martinis[1,5*]

The promise of quantum computers is that certain computational tasks might be executed exponentially faster on a quantum processor than on a classical processor[1]. A fundamental challenge is to build a high-fidelity processor capable of running quantum algorithms in an exponentially large computational space. Here we report the use of a processor with programmable superconducting qubits[2–7] to create quantum states on 53 qubits, corresponding to a computational state-space of dimension 2^{53} (about 10^{16}). Measurements from repeated experiments sample the resulting probability distribution, which we verify using classical simulations. Our Sycamore processor takes about 200 seconds to sample one instance of a quantum circuit a million times—our benchmarks currently indicate that the equivalent task for a state-of-the-art classical

Google, 2019.

Quantum Advantage

Article

Quantum supremacy using a programmable superconducting processor

Google, 2019.

RESEARCH

QUANTUM COMPUTING

Quantum computational advantage using photons

Han-Sen Zhong[1,2*], Hui Wang[1,2*], Yu-Hao Deng[1,2*], Ming-Cheng Chen[1,2*], Li-Chao Peng[1,2],

of samples (for example, ~10^{10}), the quantum advantage would be reversed if there were sufficient storage. This sample size dependence of the comparison—an analog to loopholes in Bell tests (*10*)—suggests that quantum advan-

USTC, 2021.

Strong quantum computational advantage using a superconducting quantum processor

Yulin Wu,[1,2,3] Wan-Su Bao,[4] Sirui Cao,[1,2,3] Fusheng Chen,[1,2,3] Ming-Cheng Chen,[1,2,3] Xiawei Chen,[2] Tung-Hsun Chung,[1,2,3] Hui Deng,[1,2,3] Yajie Du,[2] Daojin Fan,[1,2,3] Ming Gong,[1,2,3] Cheng Guo,[1,2,3] Chu Guo,[1,2,3] Shaojun Guo,[1,2,3] Lianchen Han,[1,2,3] Linyin Hong,[5] He-Liang Huang,[1,2,3,4] Yong-Heng Huo,[1,2,3] Liping Li,[2] Na Li,[1,2,3] Shaowei Li,[1,2,3] Yuan Li,[1,2,3] Futian Liang,[1,2,3] Chun Lin,[6] Jin Lin,[1,2,3] Haoran Qian,[1,2,3] Dan Qiao,[2] Hao Rong,[1,2,3] Hong Su,[1,2,3] Lihua Sun,[1,2,3] Liangyuan Wang,[2] Shiyu Wang,[1,2,3] Dachao Wu,[1,2,3] Yu Xu,[1,2,3] Kai Yan,[2] Weifeng Yang,[5] Yang Yang,[2] Yangsen Ye,[1,2,3] Jianghan Yin,[2] Chong Ying,[1,2,3] Jiale Yu,[1,2,3] Chen Zha,[1,2,3] Cha Zhang,[1,2,3] Haibin Zhang,[2] Kaili Zhang,[1,2,3] Yiming Zhang,[1,2,3] Han Zhao,[2] Youwei Zhao,[1,2,3] Liang Zhou,[5] Qingling Zhu,[1,2,3] Chao-Yang Lu,[1,2,3] Cheng-Zhi Peng,[1,2,3] Xiaobo Zhu,[1,2,3] and Jian-Wei Pan[1,2,3]

[1]*Hefei National Laboratory for Physical Sciences at the Microscale and Department of Modern Physics, University of Science and Technology of China, Hefei 230026, China*

USTC, 2021.

Quantum Computational Advantage via 60-Qubit 24-Cycle Random Circuit Sampling

Qingling Zhu,[1,2,3] Sirui Cao,[1,2,3] Fusheng Chen,[1,2,3] Ming-Cheng Chen,[1,2,3] Xiawei Chen,[2] Tung-Hsun Chung,[1,2,3] Hui Deng,[1,2,3] Yajie Du,[2] Daojin Fan,[1,2,3] Ming Gong,[1,2,3] Cheng Guo,[1,2,3] Chu Guo,[1,2,3] Shaojun Guo,[1,2,3] Lianchen Han,[1,2,3] Linyin Hong,[4] He-Liang Huang,[1,2,3,5] Yong-Heng Huo,[1,2,3] Liping Li,[2] Na Li,[1,2,3] Shaowei Li,[1,2,3] Yuan Li,[1,2,3] Futian Liang,[1,2,3] Chun Lin,[6] Jin Lin,[1,2,3] Haoran Qian,[1,2,3] Dan Qiao,[2] Hao Rong,[1,2,3] Hong Su,[1,2,3] Lihua Sun,[1,2,3] Liangyuan Wang,[2] Shiyu Wang,[1,2,3] Dachao Wu,[1,2,3] Yulin Wu,[1,2,3] Yu Xu,[1,2,3] Kai Yan,[2] Weifeng Yang,[4] Yang Yang,[2] Yangsen Ye,[1,2,3] Jianghan Yin,[2] Chong Ying,[1,2,3] Jiale Yu,[1,2,3] Chen Zha,[1,2,3] Cha Zhang,[1,2,3] Haibin Zhang,[2] Kaili Zhang,[1,2,3] Yiming Zhang,[1,2,3] Han Zhao,[2] Youwei Zhao,[1,2,3] Liang Zhou,[4] Chao-Yang Lu,[1,2,3] Cheng-Zhi Peng,[1,2,3] Xiaobo Zhu,[1,2,3] and Jian-Wei Pan[1,2,3]

[1]*Hefei National Laboratory for Physical Sciences at the Microscale and Department of Modern Physics, University of Science and Technology of China, Hefei 230026, China*
[2]*Shanghai Branch, CAS Center for Excellence in Quantum Information and Quantum Physics, University of Science and Technology of China, Shanghai 201315, China*
[3]*Shanghai Research Center for Quantum Sciences, Shanghai 201315, China*
[4]*QuantumCTek Co., Ltd., Hefei 230026, China*
[5]*Henan Key Laboratory of Quantum Information and Cryptography, Zhengzhou 450000, China*
[6]*Shanghai Institute of Technical Physics, Chinese Academy of Sciences, Shanghai 200083, China*

USTC, 2021.

Quantum Advantage

Quantum advantage but classically intractable to verify results.

Article

Quantum supremacy using a programmable superconducting processor

Google, 2019.

RESEARCH

QUANTUM COMPUTING

Quantum computational advantage using photons

Han-Sen Zhong[1,2]*, Hui Wang[1,2]*, Yu-Hao Deng[1,2]*, Ming-Cheng Chen[1,2]*, Li-Chao Peng[1,2],

of samples (for example, ~10^{10}), the quantum advantage would be reversed if there were sufficient storage. This sample size dependence of the comparison—an analog to loopholes in Bell tests (*10*)—suggests that quantum advan-

USTC, 2021.

Strong quantum computational advantage using a superconducting quantum processor

Yulin Wu,[1,2,3] Wan-Su Bao,[4] Sirui Cao,[1,2,3] Fusheng Chen,[1,2,3] Ming-Cheng Chen,[1,2,3] Xiawei Chen,[2] Tung-Hsun Chung,[1,2,3] Hui Deng,[1,2,3] Yajie Du,[2] Daojin Fan,[1,2,3] Ming Gong,[1,2,3] Cheng Guo,[1,2,3] Chu Guo,[1,2,3] Shaojun Guo,[1,2,3] Lianchen Han,[1,2,3] Linyin Hong,[5] He-Liang Huang,[1,2,3,4] Yong-Heng Huo,[1,2,3] Liping Li,[2] Na Li,[1,2,3] Shaowei Li,[1,2,3] Yuan Li,[1,2,3] Futian Liang,[1,2,3] Chun Lin,[6] Jin Lin,[1,2,3] Haoran Qian,[1,2,3] Dan Qiao,[2] Hao Rong,[1,2,3] Hong Su,[1,2,3] Lihua Sun,[1,2,3] Liangyuan Wang,[2] Shiyu Wang,[1,2,3] Dachao Wu,[1,2,3] Yu Xu,[1,2,3] Kai Yan,[2] Weifeng Yang,[5] Yang Yang,[2] Yangsen Ye,[1,2,3] Jianghan Yin,[2] Chong Ying,[1,2,3] Jiale Yu,[1,2,3] Chen Zha,[1,2,3] Cha Zhang,[1,2,3] Haibin Zhang,[2] Kaili Zhang,[1,2,3] Yiming Zhang,[1,2,3] Han Zhao,[2] Youwei Zhao,[1,2,3] Liang Zhou,[5] Qingling Zhu,[1,2,3] Chao-Yang Lu,[1,2,3] Cheng-Zhi Peng,[1,2,3] Xiaobo Zhu,[1,2,3] and Jian-Wei Pan[1,2,3]

[1]*Hefei National Laboratory for Physical Sciences at the Microscale and Department of Modern Physics, University of Science and Technology of China, Hefei 230026, China*

USTC, 2021.

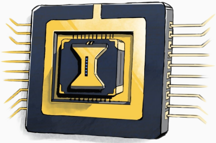
Quantum Computational Advantage via 60-Qubit 24-Cycle Random Circuit Sampling

Qingling Zhu,[1,2,3] Sirui Cao,[1,2,3] Fusheng Chen,[1,2,3] Ming-Cheng Chen,[1,2,3] Xiawei Chen,[2] Tung-Hsun Chung,[1,2,3] Hui Deng,[1,2,3] Yajie Du,[2] Daojin Fan,[1,2,3] Ming Gong,[1,2,3] Cheng Guo,[1,2,3] Chu Guo,[1,2,3] Shaojun Guo,[1,2,3] Lianchen Han,[1,2,3] Linyin Hong,[4] He-Liang Huang,[1,2,3,5] Yong-Heng Huo,[1,2,3] Liping Li,[2] Na Li,[1,2,3] Shaowei Li,[1,2,3] Yuan Li,[1,2,3] Futian Liang,[1,2,3] Chun Lin,[6] Jin Lin,[1,2,3] Haoran Qian,[1,2,3] Dan Qiao,[2] Hao Rong,[1,2,3] Hong Su,[1,2,3] Lihua Sun,[1,2,3] Liangyuan Wang,[2] Shiyu Wang,[1,2,3] Dachao Wu,[1,2,3] Yulin Wu,[1,2,3] Yu Xu,[1,2,3] Kai Yan,[2] Weifeng Yang,[4] Yang Yang,[2] Yangsen Ye,[1,2,3] Jianghan Yin,[2] Chong Ying,[1,2,3] Jiale Yu,[1,2,3] Chen Zha,[1,2,3] Cha Zhang,[1,2,3] Haibin Zhang,[2] Kaili Zhang,[1,2,3] Yiming Zhang,[1,2,3] Han Zhao,[2] Youwei Zhao,[1,2,3] Liang Zhou,[4] Chao-Yang Lu,[1,2,3] Cheng-Zhi Peng,[1,2,3] Xiaobo Zhu,[1,2,3] and Jian-Wei Pan[1,2,3]

[1]*Hefei National Laboratory for Physical Sciences at the Microscale and Department of Modern Physics, University of Science and Technology of China, Hefei 230026, China*
[2]*Shanghai Branch, CAS Center for Excellence in Quantum Information and Quantum Physics, University of Science and Technology of China, Shanghai 201315, China*
[3]*Shanghai Research Center for Quantum Sciences, Shanghai 201315, China*
[4]*QuantumCTek Co., Ltd., Hefei 230026, China*
[5]*Henan Key Laboratory of Quantum Information and Cryptography, Zhengzhou 450000, China*
[6]*Shanghai Institute of Technical Physics, Chinese Academy of Sciences, Shanghai 200083, China*

USTC, 2021.

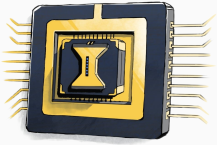
Proofs of quantumness



[0]Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Verifier

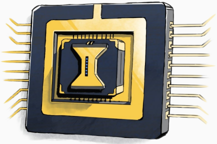


Prover

[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Verifier

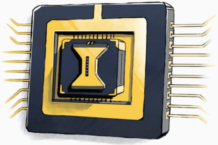


Prover

[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Verifier

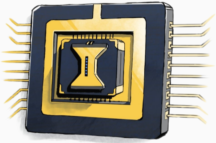


Prover

[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

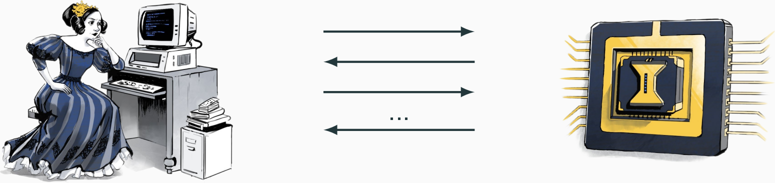
Verifier



Prover

[0]Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

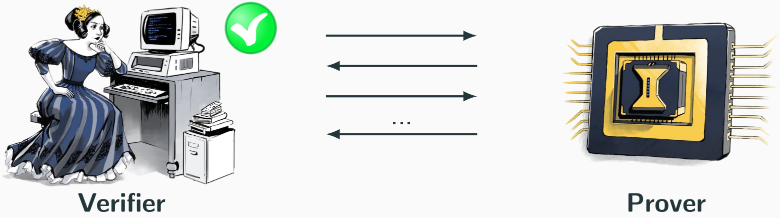


Verifier Prover

---

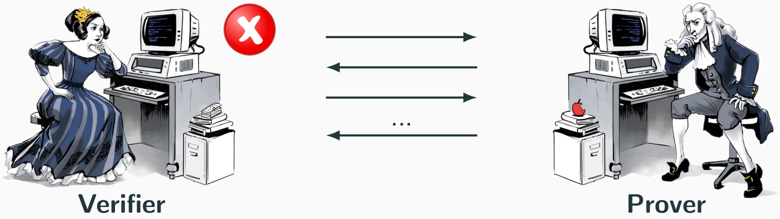
[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness



[0]Thanks to Vivian Uhlir for the figures!

Proofs of quantumness



[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Proof of quantumness (PoQ)

Let $\lambda \in \mathbb{N}$ be a security parameter. A PoQ is an interactive protocol between a $\mathrm{poly}(\lambda)$-time *classical verifier* and a $\mathrm{poly}(\lambda)$-time prover, such that

[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Proof of quantumness (PoQ)

Let $\lambda \in \mathbb{N}$ be a security parameter. A PoQ is an interactive protocol between a $\mathrm{poly}(\lambda)$-time *classical verifier* and a $\mathrm{poly}(\lambda)$-time prover, such that

- **Completeness:** There exists a *quantum prover* that makes the verifier accept with probability at least $c(\lambda)$,

[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Proof of quantumness (PoQ)

Let $\lambda \in \mathbb{N}$ be a security parameter. A PoQ is an interactive protocol between a $\mathrm{poly}(\lambda)$-time *classical verifier* and a $\mathrm{poly}(\lambda)$-time prover, such that

- **Completeness:** There exists a *quantum prover* that makes the verifier accept with probability at least $c(\lambda)$,
- **Soundness:** Any *classical prover* makes the verifier accept with probability at most $s(\lambda)$,

[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Proof of quantumness (PoQ)

Let $\lambda \in \mathbb{N}$ be a security parameter. A PoQ is an interactive protocol between a $\mathrm{poly}(\lambda)$-time *classical verifier* and a $\mathrm{poly}(\lambda)$-time prover, such that

- **Completeness:** There exists a *quantum prover* that makes the verifier accept with probability at least $c(\lambda)$,
- **Soundness:** Any *classical prover* makes the verifier accept with probability at most $s(\lambda)$,

such that $c(\lambda) - s(\lambda) > 1/\mathrm{poly}(\lambda)$.

[0] Thanks to Vivian Uhlir for the figures!

Proofs of quantumness

Proof of quantumness (PoQ)

Let $\lambda \in \mathbb{N}$ be a security parameter. A PoQ is an interactive protocol between a $\mathrm{poly}(\lambda)$-time *classical verifier* and a $\mathrm{poly}(\lambda)$-time prover, such that

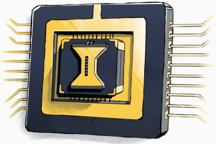
- **Completeness:** There exists a *quantum prover* that makes the verifier accept with probability at least $c(\lambda)$,
- **Soundness:** Any *classical prover* makes the verifier accept with probability at most $s(\lambda)$,

such that $c(\lambda) - s(\lambda) > 1/\mathrm{poly}(\lambda)$.

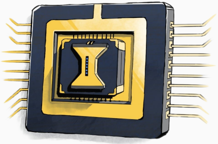
Soundness is based on a computational assumption.

[0] Thanks to Vivian Uhlir for the figures!

A simple 2-message PoQ



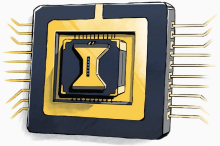
A simple 2-message PoQ



- Pick random λ-bit primes p, q and compute $N = p \cdot q$.

A simple 2-message PoQ

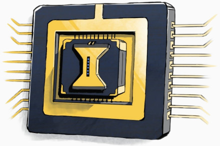
N



- Pick random λ-bit primes p, q and compute $N = p \cdot q$.
- Send N to prover.

A simple 2-message PoQ

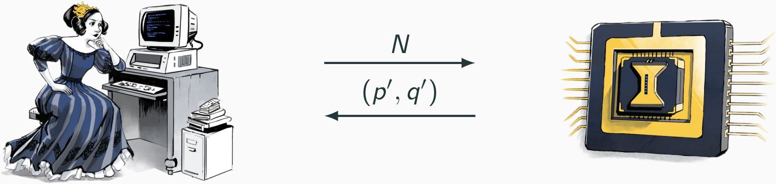
N →



- Pick random λ-bit primes p, q and compute $N = p \cdot q$.
- Send N to prover.

- Factor N using Shor's algorithm.

A simple 2-message PoQ



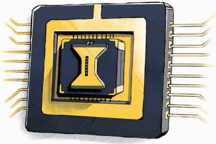
- Pick random λ-bit primes p, q and compute $N = p \cdot q$.
- Send N to prover.

- Factor N using Shor's algorithm.
- Send factors p', q' to verifier.

A simple 2-message PoQ

N

(p', q')



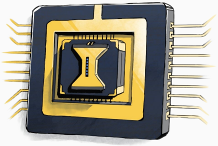
- Pick random λ-bit primes p, q and compute $N = p \cdot q$.
- Send N to prover.
- Accept if $N = p' \cdot q'$.

- Factor N using Shor's algorithm.
- Send factors p', q' to verifier.

A simple 2-message PoQ

$N \longrightarrow$

$\longleftarrow (p', q')$



- Pick random λ-bit primes p, q and compute $N = p \cdot q$.
- Send N to prover.
- Accept if $N = p' \cdot q'$.

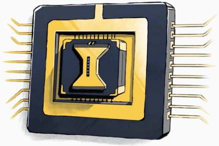
- Factor N using Shor's algorithm.
- Send factors p', q' to verifier.

PoQ, assuming *Factoring* $\notin$ BPP.

A simple 2-message PoQ

$N \longrightarrow$

$(p', q') \longleftarrow$



- Pick random λ-bit primes p, q and compute $N = p \cdot q$.
- Send N to prover.
- Accept if $N = p' \cdot q'$.

- Factor N using Shor's algorithm.
- Send factors p', q' to verifier.

PoQ, assuming *Factoring* $\notin$ BPP.

Can construct such PoQs from any problem, P, such that[1]
$P \in \text{BQP}$, $P \notin \text{BPP}$.

[1]Technically, want $P \notin$ AVBPP.

PoQs with more than 2 messages

Possible to base PoQs on some problem, P, such that $P \notin \text{BQP}$.

PoQs with more than 2 messages

Possible to base PoQs on some problem, P, such that $P \notin \mathrm{BQP}$.

PoQs can be based on the existence of *trapdoor claw-free functions.*[2]

[2] [Brakerski, Christiano, Mahadev, Vidick, Vazirani '18]

PoQs with more than 2 messages

Possible to base PoQs on some problem, P, such that $P \notin \text{BQP}$.
PoQs can be based on the existence of *trapdoor claw-free functions.*[2]

Trapdoor claw-free function (TCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a TCF family if:

[2] [Brakerski, Christiano, Mahadev, Vidick, Vazirani '18]

PoQs with more than 2 messages

Possible to base PoQs on some problem, P, such that $P \notin \mathrm{BQP}$.
PoQs can be based on the existence of *trapdoor claw-free functions.*[2]

Trapdoor claw-free function (TCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a TCF family if:

Efficient evaluation

Poly-time algorithm that, given $x \in \mathcal{I}$, computes $f_\lambda(x)$.

[2] [Brakerski, Christiano, Mahadev, Vidick, Vazirani '18]

PoQs with more than 2 messages

Possible to base PoQs on some problem, P, such that $P \notin \mathsf{BQP}$.
PoQs can be based on the existence of *trapdoor claw-free functions.*[2]

Trapdoor claw-free function (TCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a TCF family if:

Efficient evaluation

Poly-time algorithm that, given $x \in \mathcal{I}$, computes $f_\lambda(x)$.

Two-to-one

For every $y \in Im(f_\lambda)$, there are *exactly two* $x_0, x_1 \in \mathcal{I}$, $f_\lambda(x_0) = f_\lambda(x_1) = y$.

[2] [Brakerski, Christiano, Mahadev, Vidick, Vazirani '18]

PoQs with more than 2 messages

Possible to base PoQs on some problem, P, such that $P \notin \text{BQP}$.
PoQs can be based on the existence of *trapdoor claw-free functions.*[2]

Trapdoor claw-free function (TCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a TCF family if:

Efficient evaluation

Poly-time algorithm that, given $x \in \mathcal{I}$, computes $f_\lambda(x)$.

Two-to-one

For every $y \in Im(f_\lambda)$, there are *exactly two* $x_0, x_1 \in \mathcal{I}$, $f_\lambda(x_0) = f_\lambda(x_1) = y$.

Claw-free

Intractable to find $x_0, x_1 \in \mathcal{I}$, $f_\lambda(x_0) = f_\lambda(x_1) = y$.

[2] [Brakerski, Christiano, Mahadev, Vidick, Vazirani '18]

PoQs with more than 2 messages

Possible to base PoQs on some problem, P, such that $P \notin \text{BQP}$.
PoQs can be based on the existence of *trapdoor claw-free functions.*[2]

Trapdoor claw-free function (TCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a TCF family if:

Efficient evaluation

Poly-time algorithm that, given $x \in \mathcal{I}$, computes $f_\lambda(x)$.

Two-to-one

For every $y \in Im(f_\lambda)$, there are *exactly two* $x_0, x_1 \in \mathcal{I}$, $f_\lambda(x_0) = f_\lambda(x_1) = y$.

Claw-free

Intractable to find $x_0, x_1 \in \mathcal{I}$, $f_\lambda(x_0) = f_\lambda(x_1) = y$.

Trapdoor

There is a trapdoor t_λ and a poly-time algorithm that, given t_λ and $y \in Im(f_\lambda)$ can compute $x_0, x_1 \in \mathcal{I}$, such that $f_\lambda(x_0) = f_\lambda(x_1) = y$.

[2][Brakerski, Christiano, Mahadev, Vidick, Vazirani '18]

PoQs with more than 2 messages

Strong trapdoor claw-free function (STCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a STCF family if it is a TCF and:

PoQs with more than 2 messages

Strong trapdoor claw-free function (STCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a STCF family if it is a TCF and:

Adaptive hardcore bit

Intractable to find $y \in Im(f_\lambda)$, $x_0 \in \mathcal{I}$, and $d \in \{0,1\}^{\mathrm{poly}(\lambda)}$ ($d \neq 0$) such that:

PoQs with more than 2 messages

Strong trapdoor claw-free function (STCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a STCF family if it is a TCF and:

Adaptive hardcore bit

Intractable to find $y \in Im(f_\lambda)$, $x_0 \in \mathcal{I}$, and $d \in \{0,1\}^{\mathrm{poly}(\lambda)}$ ($d \neq 0$) such that:

$$d \cdot (x_0 \oplus x_1) = 0,$$
$$f_\lambda(x_0) = f_\lambda(x_1) = y,$$

with probability non-negligibly greater than 1/2.

PoQs with more than 2 messages

Strong trapdoor claw-free function (STCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a STCF family if it is a TCF and:

Adaptive hardcore bit

Intractable to find $y \in Im(f_\lambda)$, $x_0 \in \mathcal{I}$, and $d \in \{0,1\}^{\mathrm{poly}(\lambda)}$ ($d \neq 0$) such that:

$$d \cdot (x_0 \oplus x_1) = 0,$$
$$f_\lambda(x_0) = f_\lambda(x_1) = y,$$

with probability non-negligibly greater than 1/2.

Intuition: if you know x_0 you shouldn't know even a single bit of x_1.

PoQs with more than 2 messages

Strong trapdoor claw-free function (STCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a STCF family if it is a TCF and:

Adaptive hardcore bit

Intractable to find $y \in Im(f_\lambda)$, $x_0 \in \mathcal{I}$, and $d \in \{0,1\}^{\mathrm{poly}(\lambda)}$ ($d \neq 0$) such that:

$$d \cdot (x_0 \oplus x_1) = 0,$$
$$f_\lambda(x_0) = f_\lambda(x_1) = y,$$

with probability non-negligibly greater than 1/2.

Intuition: if you know x_0 you shouldn't know even a single bit of x_1.

Adaptive hardcore bit implies claw-free property.

PoQs with more than 2 messages

Strong trapdoor claw-free function (STCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a STCF family if it is a TCF and:

Adaptive hardcore bit

Intractable to find $y \in Im(f_\lambda)$, $x_0 \in \mathcal{I}$, and $d \in \{0,1\}^{\mathrm{poly}(\lambda)}$ ($d \neq 0$) such that:

$$d \cdot (x_0 \oplus x_1) = 0,$$
$$f_\lambda(x_0) = f_\lambda(x_1) = y,$$

with probability non-negligibly greater than 1/2.

Intuition: if you know x_0 you shouldn't know even a single bit of x_1.

Adaptive hardcore bit implies claw-free property.

STCFs can be constructed from LWE.

PoQs with more than 2 messages

Strong trapdoor claw-free function (STCF)

We say a family $\{f_\lambda : \mathcal{I} \to \mathcal{O}\}_{\lambda \in \mathbb{N}}$ is a STCF family if it is a TCF and:

Adaptive hardcore bit

Intractable to find $y \in Im(f_\lambda)$, $x_0 \in \mathcal{I}$, and $d \in \{0,1\}^{\mathrm{poly}(\lambda)}$ ($d \neq 0$) such that:

$$d \cdot (x_0 \oplus x_1) = 0,$$
$$f_\lambda(x_0) = f_\lambda(x_1) = y,$$

with probability non-negligibly greater than 1/2.

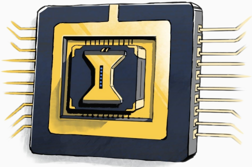
Intuition: if you know x_0 you shouldn't know even a single bit of x_1.

Adaptive hardcore bit implies claw-free property.

STCFs can be constructed from LWE.

TCFs can be constructed from factoring, discrete-log, Ring-LWE, LWE.

A 4-message PoQ (the BCMVV'18 protocol)

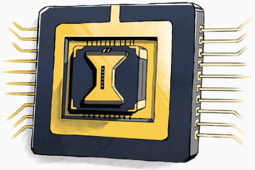


A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

t_λ

f_λ →



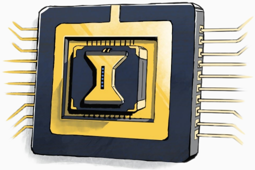
A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

t_λ

f_λ →

$y \in Im(f_\lambda)$ ←



A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

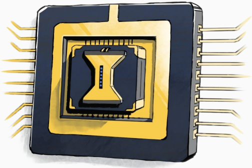
With probability 1/2.

t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

Preimage →



A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

With probability 1/2.

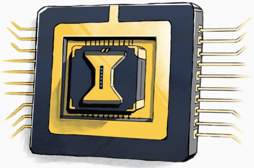
t_λ

f_λ →

← $y \in Im(f_\lambda)$

Preimage →

← x



A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

With probability 1/2.

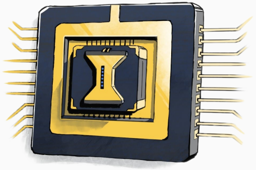
t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

Preimage →

x ←



Verifier accepts if $f_\lambda(x) = y$.

A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

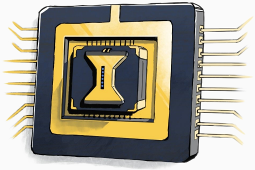
With probability 1/2.

t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

Equation →



A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

With probability 1/2.

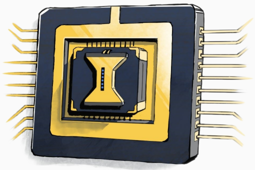
t_λ

f_λ →

← $y \in Im(f_\lambda)$

Equation →

← d



Verifier accepts if $d \cdot (x_0 \oplus x_1) = 0$, with $f_\lambda(x_0) = f_\lambda(x_1) = y$.

A 4-message PoQ (the BCMVV'18 protocol)

Verifier generates STCF, f_λ, together with trapdoor t_λ.

With probability 1/2.

Verifier uses t_λ to obtain x_0, x_1 from y and checks the equation.

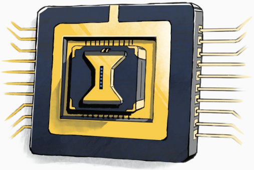
t_λ

f_λ →

← $y \in Im(f_\lambda)$

Equation →

← d



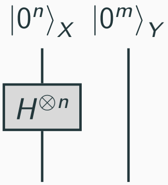
Verifier accepts if $d \cdot (x_0 \oplus x_1) = 0$, with $f_\lambda(x_0) = f_\lambda(x_1) = y$.

BCMVV'18 completeness

$|0^n\rangle_X \ |0^m\rangle_Y$

$|0^n\rangle_X |0^m\rangle_Y$

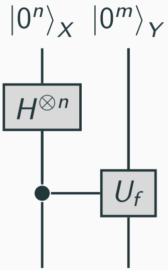
BCMVV'18 completeness



$|0^n\rangle_X |0^m\rangle_Y$

$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle_X \quad |0^m\rangle_Y$

BCMVV'18 completeness

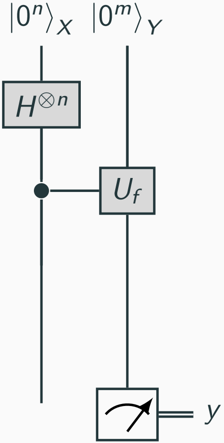


$$|0^n\rangle_X \, |0^m\rangle_Y$$

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle_X \quad |0^m\rangle_Y$$

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle_X \quad |f_\lambda(x)\rangle_Y$$

BCMVV'18 completeness



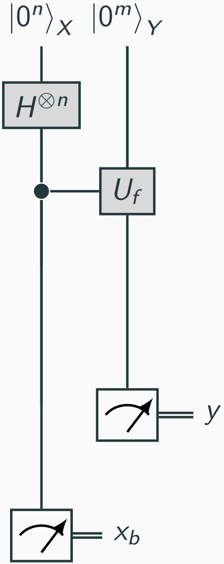
$$|0^n\rangle_X \, |0^m\rangle_Y$$

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle_X \quad |0^m\rangle_Y$$

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle_X \quad |f_\lambda(x)\rangle_Y$$

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)_X \quad |y\rangle_Y$$

BCMVV'18 completeness



$$|0^n\rangle_X\,|0^m\rangle_Y$$

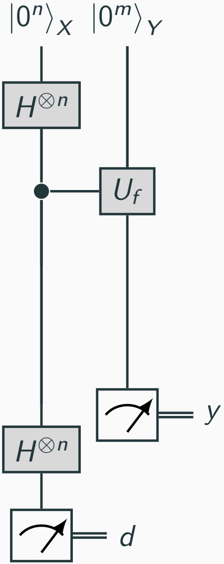
$$\frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle_X \quad |0^m\rangle_Y$$

$$\frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle_X \quad |f_\lambda(x)\rangle_Y$$

$$\frac{1}{\sqrt{2}}(|x_0\rangle+|x_1\rangle)_X \quad |y\rangle_Y$$

Preimage case: x_b, $b \leftarrow_U \{0,1\}$

BCMVV'18 completeness



$$|0^n\rangle_X |0^m\rangle_Y$$

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle_X \quad |0^m\rangle_Y$$

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle_X \quad |f_\lambda(x)\rangle_Y$$

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)_X \quad |y\rangle_Y$$

Equation case: d, $d \cdot (x_0 \oplus x_1) = 0$

BCMVV'18 soundness

BCMVV'18 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.

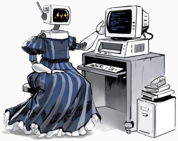
BCMVV'18 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.

Use it to construct poly-time algorithm that breaks adaptive hardcore bit.

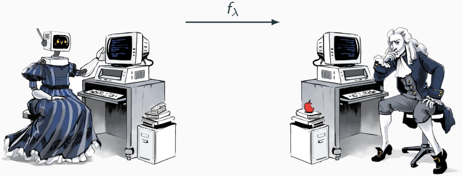
BCMVV'18 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



BCMVV'18 soundness

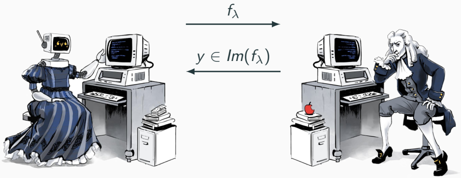
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ

BCMVV'18 soundness

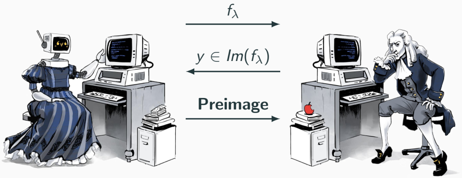
Assume there is a poly-time classical prover that succeeds in the protocol. Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image

BCMVV'18 soundness

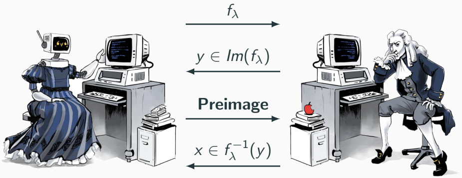
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image
3. Ask for preimage

BCMVV'18 soundness

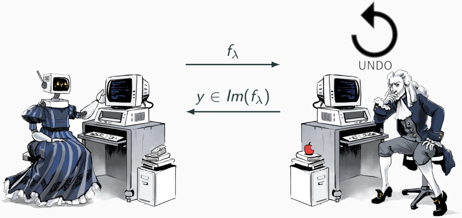
Assume there is a poly-time classical prover that succeeds in the protocol. Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image
3. Ask for preimage

BCMVV'18 soundness

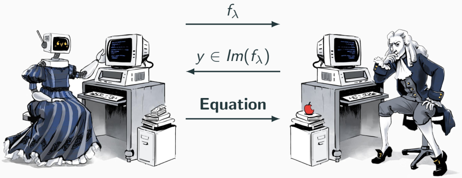
Assume there is a poly-time classical prover that succeeds in the protocol. Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**

BCMVV'18 soundness

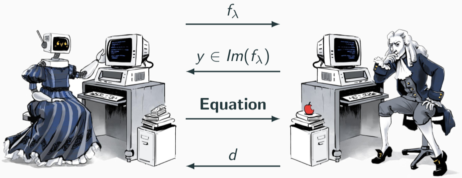
Assume there is a poly-time classical prover that succeeds in the protocol. Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Ask for equation

BCMVV'18 soundness

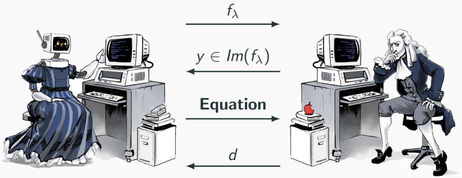
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Ask for equation

BCMVV'18 soundness

Assume there is a poly-time classical prover that succeeds in the protocol. Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



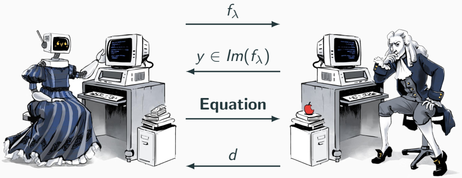
1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Ask for equation

We've constructed a poly-time algorithm that produces (y, x_b, d), with

$$d \cdot (x_0 \oplus x_1) = 0$$

BCMVV'18 soundness

Assume there is a poly-time classical prover that succeeds in the protocol. Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Ask for equation

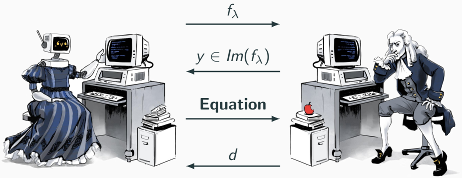
We've constructed a poly-time algorithm that produces (y, x_b, d), with

$$d \cdot (x_0 \oplus x_1) = 0$$

Contradicts adaptive hardcore bit property!

BCMVV'18 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks adaptive hardcore bit.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Ask for equation

We've constructed a poly-time algorithm that produces (y, x_b, d), with

$$d \cdot (x_0 \oplus x_1) = 0$$

Contradicts adaptive hardcore bit property!

BCMVV'18 proof of quantumness

BCMVV'18 is a 4-message PoQ with $c(\lambda) = 1$ and $s(\lambda) = 3/4 + \mathrm{negl}(\lambda)$.

Observations about BCMVV'18

- Soundness relies on the adaptive hardcore bit property of STCFs.

Observations about BCMVV'18

- Soundness relies on the adaptive hardcore bit property of STCFs.
- STCFs implemented from LWE (believed to be quantum hard!).

Observations about BCMVV'18

- Soundness relies on the adaptive hardcore bit property of STCFs.
- STCFs implemented from LWE (believed to be quantum hard!).
- Interaction allows for classical rewinding, but no quantum rewinding.

Observations about BCMVV'18

- Soundness relies on the adaptive hardcore bit property of STCFs.
- STCFs implemented from LWE (believed to be quantum hard!).
- Interaction allows for classical rewinding, but no quantum rewinding.
- Protocol can be parallel-repeated to yield $c(\lambda) = 1$, $s(\lambda) = \mathrm{negl}(\lambda)$.

Observations about BCMVV'18

- Soundness relies on the adaptive hardcore bit property of STCFs.
- STCFs implemented from LWE (believed to be quantum hard!).
- Interaction allows for classical rewinding, but no quantum rewinding.
- Protocol can be parallel-repeated to yield $c(\lambda) = 1$, $s(\lambda) = \mathrm{negl}(\lambda)$.

Is the adaptive hardcore bit *necessary*?
Can we base PoQs on just TCFs?

Observations about BCMVV'18

- Soundness relies on the adaptive hardcore bit property of STCFs.
- STCFs implemented from LWE (believed to be quantum hard!).
- Interaction allows for classical rewinding, but no quantum rewinding.
- Protocol can be parallel-repeated to yield $c(\lambda) = 1$, $s(\lambda) = \mathrm{negl}(\lambda)$.

Is the adaptive hardcore bit *necessary*?
Can we base PoQs on just TCFs?

Yes! By "forcing" an equation onto the prover[3].

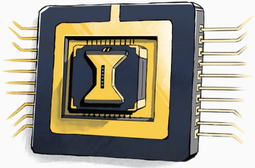
[3] [Kahanamoku-Meyer, Choi, Vazirani, Yao, 2021]

A 6-message PoQ (KCVY'21 protocol)

t_λ

f_λ

$y \in Im(f_\lambda)$

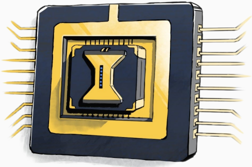


A 6-message PoQ (KCVY'21 protocol)

t_λ

f_λ

$y \in Im(f_\lambda)$



$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$

A 6-message PoQ (KCVY'21 protocol)

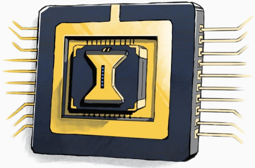
t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

Preimage →

$x \in f_\lambda^{-1}(y)$ ←



$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$

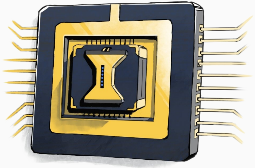
A 6-message PoQ (KCVY'21 protocol)

t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

r →



$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$

$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$

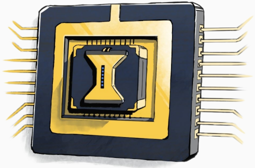
A 6-message PoQ (KCVY'21 protocol)

t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

r →



$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$

$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$

$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle |x_0\rangle + |r \cdot x_1\rangle |x_1\rangle)$

A 6-message PoQ (KCVY'21 protocol)

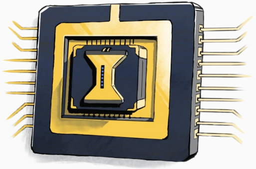
t_λ

$\xrightarrow{f_\lambda}$

$\xleftarrow{y \in Im(f_\lambda)}$

$\xrightarrow{r}$

$\xleftarrow{d}$



$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle |x_0\rangle + |r \cdot x_1\rangle |x_1\rangle)$$

Hadamard and measure second register

A 6-message PoQ (KCVY'21 protocol)

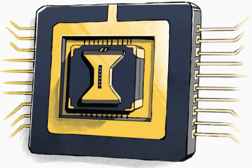
t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

r →

d ←



$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle |x_0\rangle + |r \cdot x_1\rangle |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d \cdot (x_0 \oplus x_1)} |r \cdot x_1\rangle)$$

A 6-message PoQ (KCVY'21 protocol)

t_λ

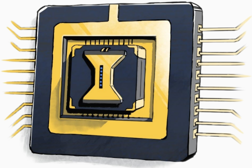
f_λ →

$y \in Im(f_\lambda)$ ←

r →

d ←

m →



$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$

$m \leftarrow_U \{-\pi/4, \pi/4\}$

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle |x_0\rangle + |r \cdot x_1\rangle |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d \cdot (x_0 \oplus x_1)}|r \cdot x_1\rangle)$$

A 6-message PoQ (KCVY'21 protocol)

t_λ

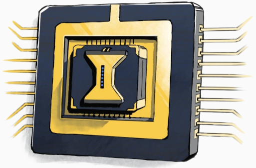
f_λ →

$y \in Im(f_\lambda)$ ←

r →

d ←

m →



$$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$$

$$m \leftarrow_U \{-\pi/4, \pi/4\}$$

$$\left\{ \begin{array}{l} \cos\left(\frac{m}{2}\right)|0\rangle + \sin\left(\frac{m}{2}\right)|1\rangle \\ \cos\left(\frac{m}{2}\right)|1\rangle - \sin\left(\frac{m}{2}\right)|0\rangle \end{array} \right\}$$

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle |x_0\rangle + |r \cdot x_1\rangle |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d \cdot (x_0 \oplus x_1)} |r \cdot x_1\rangle)$$

A 6-message PoQ (KCVY'21 protocol)

t_λ

f_λ →

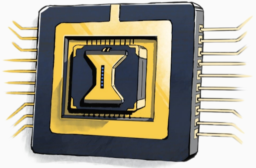
$y \in Im(f_\lambda)$ ←

r →

d ←

m →

o ←



$$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$$

$$m \leftarrow_U \{-\pi/4, \pi/4\}$$

$$\left\{ \begin{array}{ll} \cos\left(\frac{m}{2}\right)|0\rangle + & \sin\left(\frac{m}{2}\right)|1\rangle \\ \cos\left(\frac{m}{2}\right)|1\rangle - & \sin\left(\frac{m}{2}\right)|0\rangle \end{array} \right\}$$

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle|x_0\rangle + |r \cdot x_1\rangle|x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d\cdot(x_0 \oplus x_1)}|r \cdot x_1\rangle)$$

Measure in basis m, outcome o

A 6-message PoQ (KCVY'21 protocol)

t_λ

$\xrightarrow{f_\lambda}$

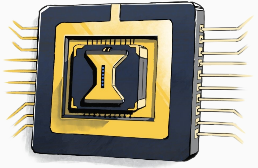
$\xleftarrow{y \in Im(f_\lambda)}$

$\xrightarrow{r}$

$\xleftarrow{d}$

$\xrightarrow{m}$

$\xleftarrow{o}$



$$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$$

$$m \leftarrow_U \{-\pi/4, \pi/4\}$$

$$\left\{ \begin{array}{ll} \cos\left(\frac{m}{2}\right)|0\rangle + & \sin\left(\frac{m}{2}\right)|1\rangle \\ \cos\left(\frac{m}{2}\right)|1\rangle - & \sin\left(\frac{m}{2}\right)|0\rangle \end{array} \right\}$$

Use t_λ, r, d to compute *likely* o.

Accept if prover sends likely o.

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle|x_0\rangle + |r \cdot x_1\rangle|x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d\cdot(x_0 \oplus x_1)}|r \cdot x_1\rangle)$$

Measure in basis m, outcome o

A 6-message PoQ (KCVY'21 protocol)

t_λ

f_λ →

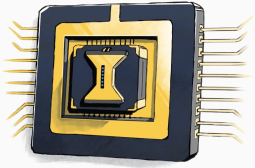
$y \in Im(f_\lambda)$ ←

r →

d ←

m →

o ←



$$r \leftarrow_U \{0,1\}^{\mathrm{poly}(\lambda)}$$

$$m \leftarrow_U \{-\pi/4, \pi/4\}$$

$$\left\{ \begin{array}{ll} \cos\left(\frac{m}{2}\right)|0\rangle + & \sin\left(\frac{m}{2}\right)|1\rangle \\ \cos\left(\frac{m}{2}\right)|1\rangle - & \sin\left(\frac{m}{2}\right)|0\rangle \end{array} \right\}$$

Use t_λ, r, d to compute *likely* o.

Accept if prover sends likely o.

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle |x_0\rangle + |r \cdot x_1\rangle |x_1\rangle)$$

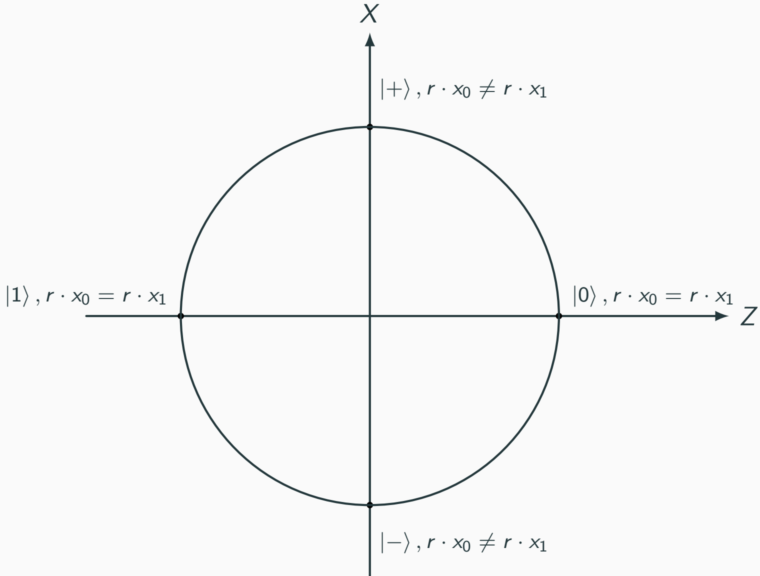
$$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d\cdot(x_0 \oplus x_1)}|r \cdot x_1\rangle)$$

Measure in basis m, outcome o

Quantum prover succeeds with probability $cos^2(\pi/8) \approx 85\%$

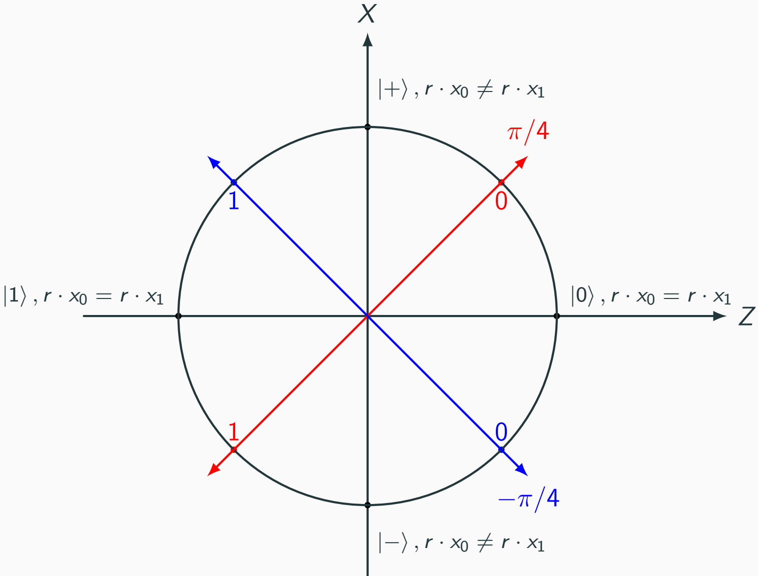
A 6-message PoQ (KCVY'21 protocol)

$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d\cdot(x_0 \oplus x_1)} |r \cdot x_1\rangle)$



A 6-message PoQ (KCVY'21 protocol)

$\frac{1}{\sqrt{2}}(|r \cdot x_0\rangle + (-1)^{d \cdot (x_0 \oplus x_1)} |r \cdot x_1\rangle)$



KCVY'21 soundness

KCVY'21 soundness

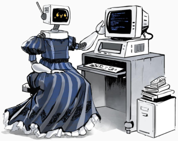
Assume there is a poly-time classical prover that succeeds in the protocol.

KCVY'21 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.

KCVY'21 soundness

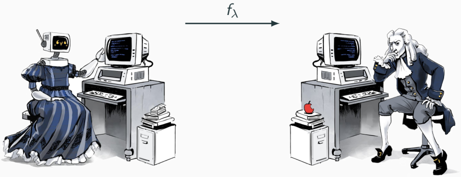
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



KCVY'21 soundness

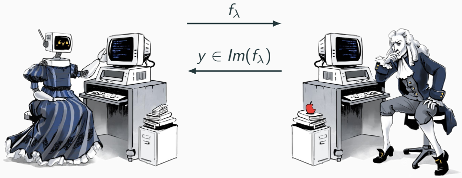
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.

1. Send f_λ



KCVY'21 soundness

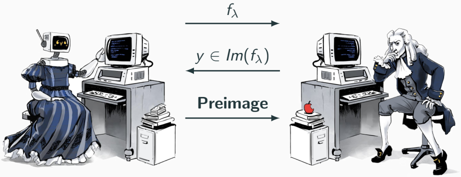
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image

KCVY'21 soundness

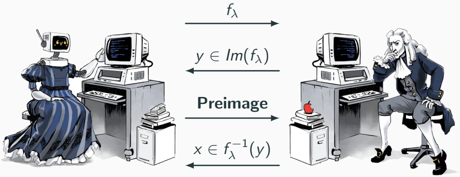
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage

KCVY'21 soundness

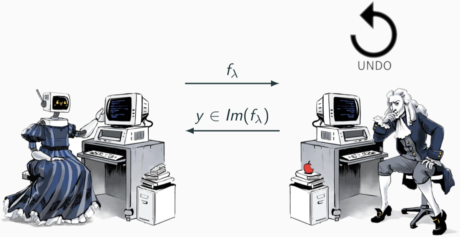
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage

KCVY'21 soundness

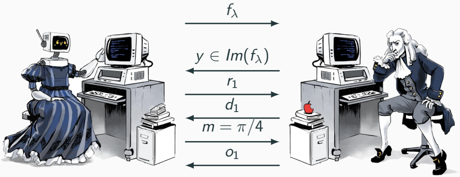
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**

KCVY'21 soundness

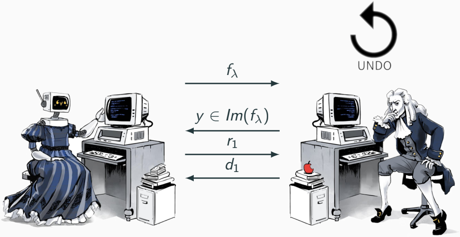
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.

KCVY'21 soundness

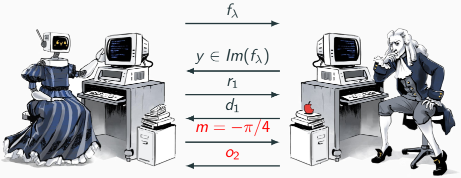
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.
6. **Rewind**

KCVY'21 soundness

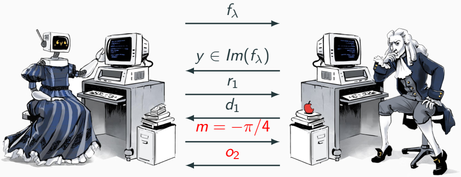
Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.
6. **Rewind**
7. Do Bell test with r_1, $m = -\pi/4$.

KCVY'21 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.

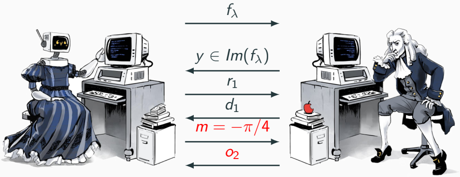


1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.
6. **Rewind**
7. Do Bell test with r_1, $m = -\pi/4$.

   Repeat with $r_2, r_3, \ldots$

KCVY'21 soundness

Assume there is a poly-time classical prover that succeeds in the protocol. Use it to construct poly-time algorithm that breaks TCF claw-freeness.



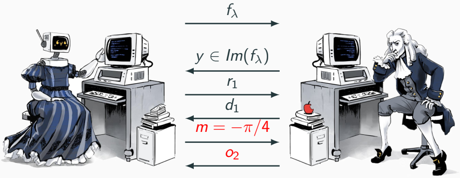
1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.
6. **Rewind**
7. Do Bell test with r_1, $m = -\pi/4$.

   Repeat with $r_2, r_3, \ldots$

Outcomes $o_1, o_2, \ldots$ determine bits of $x_0 \oplus x_1$.

KCVY'21 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.
6. **Rewind**
7. Do Bell test with r_1, $m = -\pi/4$.

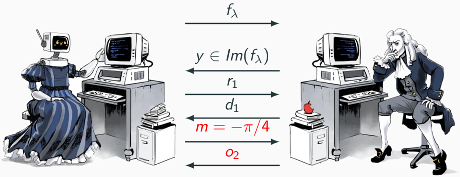
   Repeat with $r_2, r_3, \ldots$

Outcomes $o_1, o_2, \ldots$ determine bits of $x_0 \oplus x_1$.

After poly-many repetitions, can decode $x_0 \oplus x_1$, *à la* Goldreich-Levin.

KCVY'21 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.
6. **Rewind**
7. Do Bell test with r_1, $m = -\pi/4$.

   Repeat with $r_2, r_3, \ldots$

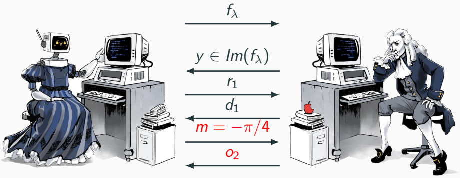
Outcomes $o_1, o_2, \ldots$ determine bits of $x_0 \oplus x_1$.

After poly-many repetitions, can decode $x_0 \oplus x_1$, *à la* Goldreich-Levin.

Can recover both preimages, which contradicts claw-freeness!

KCVY'21 soundness

Assume there is a poly-time classical prover that succeeds in the protocol.
Use it to construct poly-time algorithm that breaks TCF claw-freeness.



1. Send f_λ
2. Ask for image
3. Ask for preimage
4. **Rewind**
5. Do Bell test with r_1, $m = \pi/4$.
6. **Rewind**
7. Do Bell test with r_1, $m = -\pi/4$.

   Repeat with $r_2, r_3, \ldots$

KCVY'21 proof of quantumness

KCVY'21 is a 6-message PoQ with $c(\lambda) = \frac{1}{2}(1 + cos^2(\pi/8))$ and $s(\lambda) = \frac{1}{2}(1 + 3/4) + \mathrm{negl}(\lambda)$.

KCVY'21 removing the preimage test

Brakerski, Porat and Vidick showed that preimage test can be removed!

KCVY'21 removing the preimage test

Brakerski, Porat and Vidick showed that preimage test can be removed!

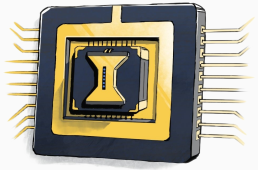
There's another way to do this with a simple modification...

KCVY'21 removing the preimage test

t_λ

f_λ →

$y \in Im(f_\lambda)$ ←



$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$

[3] [Gheorghiu, Kahanamoku-Meyer]

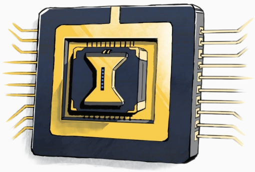
KCVY'21 removing the preimage test

t_λ

f_λ →

$y \in Im(f_\lambda)$ ←

r →



$\frac{1}{\sqrt{2}}(|x_0, 00..0\rangle + |00..0, x_1\rangle)$

[3] [Gheorghiu, Kahanamoku-Meyer]

KCVY'21 removing the preimage test

t_λ

f_λ →

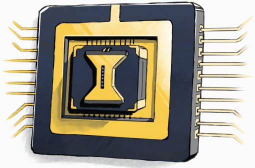
$y \in Im(f_\lambda)$ ←

r →

d ←

m →

o ←



$r \leftarrow_U \{0,1\}^{2\text{poly}(\lambda)}$

$m \leftarrow_U \{-\pi/4, \pi/4\}$

Use t_λ, r, d to compute *likely* o.

Accept if prover sends likely o.

$$\frac{1}{\sqrt{2}}(|x_0, 00..0\rangle + |00..0, x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot (x_0, 00..0)\rangle + (-1)^{d\cdot(x_0 \oplus x_1)}|r \cdot (00..0, x_1)\rangle)$$

Measure in basis m, outcome o

[3] [Gheorghiu, Kahanamoku-Meyer]

KCVY'21 removing the preimage test

t_λ

f_λ →

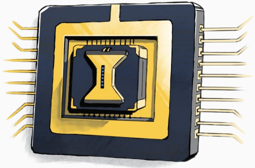
$y \in Im(f_\lambda)$ ←

r →

d ←

m →

o ←



$r \leftarrow_U \{0,1\}^{2\mathrm{poly}(\lambda)}$

$m \leftarrow_U \{-\pi/4, \pi/4\}$

Use t_λ, r, d to compute *likely* o.

Accept if prover sends likely o.

$\frac{1}{\sqrt{2}}(|x_0, 00..0\rangle + |00..0, x_1\rangle)$

$\frac{1}{\sqrt{2}}(|r \cdot (x_0, 00..0)\rangle + (-1)^{d\cdot(x_0 \oplus x_1)}|r \cdot (00..0, x_1)\rangle)$

Measure in basis m, outcome o

Hardcore bit is now $r \cdot (x_0||x_1)$.

[3] [Gheorghiu, Kahanamoku-Meyer]

KCVY'21 removing the preimage test

t_λ

f_λ →

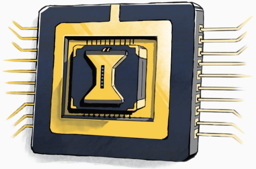
← $y \in Im(f_\lambda)$

r →

← d

m →

← o



$r \leftarrow_U \{0,1\}^{2\text{poly}(\lambda)}$

$m \leftarrow_U \{-\pi/4, \pi/4\}$

Use t_λ, r, d to compute *likely* o.

Accept if prover sends likely o.

$$\frac{1}{\sqrt{2}}(|x_0, 00..0\rangle + |00..0, x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot (x_0, 00..0)\rangle + (-1)^{d\cdot(x_0 \oplus x_1)}|r \cdot (00..0, x_1)\rangle)$$

Measure in basis m, outcome o

Hardcore bit is now $r \cdot (x_0 || x_1)$.

When doing the decoding in the soundness analysis, recover $x_0||x_1$.

[3] [Gheorghiu, Kahanamoku-Meyer]

KCVY'21 removing the preimage test

t_λ

f_λ →

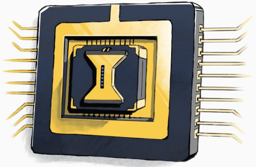
$y \in Im(f_\lambda)$ ←

r →

d ←

m →

o ←



$r \leftarrow_U \{0,1\}^{2\mathrm{poly}(\lambda)}$

$m \leftarrow_U \{-\pi/4, \pi/4\}$

Use t_λ, r, d to compute *likely* o.

Accept if prover sends likely o.

$$\frac{1}{\sqrt{2}}(|x_0, 00..0\rangle + |00..0, x_1\rangle)$$

$$\frac{1}{\sqrt{2}}(|r \cdot (x_0, 00..0)\rangle + (-1)^{d\cdot(x_0 \oplus x_1)}|r \cdot (00..0, x_1)\rangle)$$

Measure in basis m, outcome o

Preimageless KCVY'21 proof of quantumness

6-message PoQ with $c(\lambda) = cos^2(\pi/8)$ and $s(\lambda) = 3/4 + \mathrm{negl}(\lambda)$.

[3] [Gheorghiu, Kahanamoku-Meyer], [Brakerski, Porat, Vidick]

Observations about KCVY'21

- Soundness relies only on claw-free property of TCFs.

Observations about KCVY'21

- Soundness relies only on claw-free property of TCFs.
- Introduces additional round of interaction, with respect to BCMVV'18 (6 vs 4 messages).

Observations about KCVY'21

- Soundness relies only on claw-free property of TCFs.
- Introduces additional round of interaction, with respect to BCMVV'18 (6 vs 4 messages).
- Multiple rewindings required (compared to single rewinding in BCMVV'18).

Observations about KCVY'21

- Soundness relies only on claw-free property of TCFs.
- Introduces additional round of interaction, with respect to BCMVV'18 (6 vs 4 messages).
- Multiple rewindings required (compared to single rewinding in BCMVV'18).
- TCFs can be constructed from multiple crypto problems (factoring, discrete-log, ring-LWE, LWE).

Observations about KCVY'21

- Soundness relies only on claw-free property of TCFs.
- Introduces additional round of interaction, with respect to BCMVV'18 (6 vs 4 messages).
- Multiple rewindings required (compared to single rewinding in BCMVV'18).
- TCFs can be constructed from multiple crypto problems (factoring, discrete-log, ring-LWE, LWE).
- **Key point:** quantum strategy in KCVY'21 with a factoring-based TCF is much simpler than performing Shor's algorithm!

Observations about KCVY'21

- Soundness relies only on claw-free property of TCFs.
- Introduces additional round of interaction, with respect to BCMVV'18 (6 vs 4 messages).
- Multiple rewindings required (compared to single rewinding in BCMVV'18).
- TCFs can be constructed from multiple crypto problems (factoring, discrete-log, ring-LWE, LWE).
- **Key point:** quantum strategy in KCVY'21 with a factoring-based TCF is much simpler than performing Shor's algorithm!
- Requires "only" $2\lambda + 1$ qubits and $O(\lambda \log(\lambda))$ gates (compared to $O(\lambda^3)$ gates for Shor's algorithm).

Observations about KCVY'21

- Soundness relies only on claw-free property of TCFs.
- Introduces additional round of interaction, with respect to BCMVV'18 (6 vs 4 messages).
- Multiple rewindings required (compared to single rewinding in BCMVV'18).
- TCFs can be constructed from multiple crypto problems (factoring, discrete-log, ring-LWE, LWE).
- **Key point:** quantum strategy in KCVY'21 with a factoring-based TCF is much simpler than performing Shor's algorithm!
- Requires "only" $2\lambda + 1$ qubits and $O(\lambda \log(\lambda))$ gates (compared to $O(\lambda^3)$ gates for Shor's algorithm).

Potential for performing PoQs with non-fault tolerant quantum devices[4]...

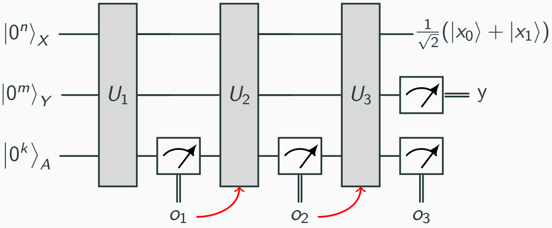
[4] Interactive Protocols for Classically-Verifiable Quantum Advantage, Zhu et al. '22.

Constant-depth PoQs

Possible to make quantum strategy use only constant-depth circuits
(together with log-depth classical computation)

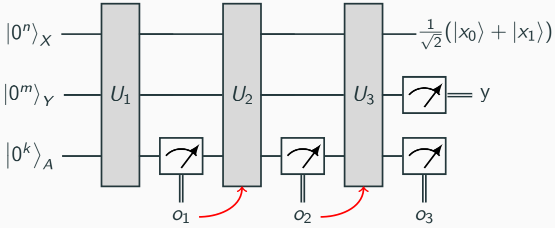
Constant-depth PoQs

Possible to make quantum strategy use only constant-depth circuits
(together with log-depth classical computation)



Constant-depth PoQs

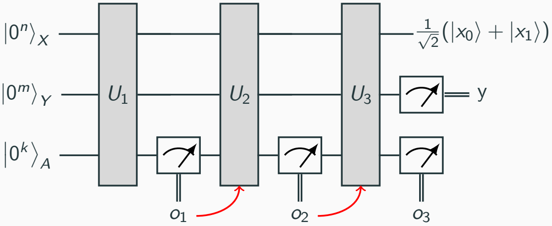
Possible to make quantum strategy use only constant-depth circuits (together with log-depth classical computation)



Where U_1, U_2, U_3 are constant-depth circuits.

Constant-depth PoQs

Possible to make quantum strategy use only constant-depth circuits (together with log-depth classical computation)

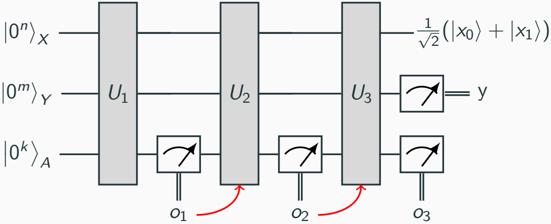


Where U_1, U_2, U_3 are constant-depth circuits.

[Hirahara, Le Gall, 2021] for STCFs based on LWE.
[Liu, Gheorghiu, 2021] for TCFs and STCFs.

Constant-depth PoQs

Possible to make quantum strategy use only constant-depth circuits
(together with log-depth classical computation)



Where U_1, U_2, U_3 are constant-depth circuits.

[Hirahara, Le Gall, 2021] for STCFs based on LWE.
[Liu, Gheorghiu, 2021] for TCFs and STCFs.

Circuit width becomes quite high $O(\lambda^8 \log(\lambda))$.

Non-interactive PoQs

Can we make PoQs non-interactive (2-message protocols)?

Non-interactive PoQs - BKVV'20

Can we make PoQs non-interactive (2-message protocols)?

Yes, in the random oracle model (ROM)[5].

[5] [Brakerski, Koppula, Vazirani, Vidick, 2020]

Non-interactive PoQs - BKVV'20

Can we make PoQs non-interactive (2-message protocols)?

Yes, in the random oracle model (ROM)[5].

ROM = Verifier and prover given access to a random function $H : \{0,1\}^{\text{poly}(\lambda)} \rightarrow \{0,1\}$.

[5] [Brakerski, Koppula, Vazirani, Vidick, 2020]

Non-interactive PoQs - BKVV'20

Can we make PoQs non-interactive (2-message protocols)?

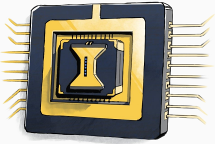
Yes, in the random oracle model (ROM)[5].

ROM = Verifier and prover given access to a random function
$H : \{0,1\}^{\mathrm{poly}(\lambda)} \to \{0,1\}$.

t_λ

f_λ →

(y, d, b) ←



[5] [Brakerski, Koppula, Vazirani, Vidick, 2020]

Non-interactive PoQs - BKVV'20

Can we make PoQs non-interactive (2-message protocols)?

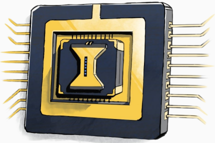
Yes, in the random oracle model (ROM)[5].

ROM = Verifier and prover given access to a random function $H : \{0,1\}^{\mathrm{poly}(\lambda)} \rightarrow \{0,1\}$.

t_λ

f_λ

(y, d, b)



$y \in Im(f), d \in \{0,1\}^{\mathrm{poly}(\lambda)}, b \in \{0,1\}$.

[5] [Brakerski, Koppula, Vazirani, Vidick, 2020]

Non-interactive PoQs - BKVV'20

Can we make PoQs non-interactive (2-message protocols)?

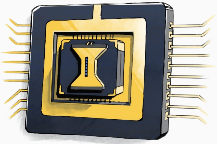
Yes, in the random oracle model (ROM)[5].

ROM = Verifier and prover given access to a random function $H : \{0,1\}^{\mathrm{poly}(\lambda)} \to \{0,1\}$.

t_λ

f_λ →

← (y, d, b)



$y \in Im(f), d \in \{0,1\}^{\mathrm{poly}(\lambda)}, b \in \{0,1\}$.

Verifier accepts if

$$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1).$$

[5] [Brakerski, Koppula, Vazirani, Vidick, 2020]

Non-interactive PoQs - BKVV'20

Completeness

Non-interactive PoQs - BKVV'20

Completeness

Prover prepares state $\frac{1}{\sqrt{2}}\left(|0\rangle|x_0\rangle + (-1)^{H(x_0)+H(x_1)}|1\rangle|x_1\rangle\right)$

Non-interactive PoQs - BKVV'20

Completeness

Prover prepares state $\frac{1}{\sqrt{2}}\left(|0\rangle|x_0\rangle + (-1)^{H(x_0)+H(x_1)}|1\rangle|x_1\rangle\right)$
Can be done by evaluating H in superposition (in addition to f_λ).

Non-interactive PoQs - BKVV'20

Completeness

Prover prepares state $\frac{1}{\sqrt{2}}\left(|0\rangle |x_0\rangle + (-1)^{H(x_0)+H(x_1)} |1\rangle |x_1\rangle\right)$

Can be done by evaluating H in superposition (in addition to f_λ).

Measuring state in Hadamard basis yields (b, d).

$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}} \left(|0\rangle |x_0\rangle + (-1)^{H(x_0)+H(x_1)} |1\rangle |x_1\rangle \right)$

Can be done by evaluating H in superposition (in addition to f_λ).

Measuring state in Hadamard basis yields (b, d).

$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}}\left(|0\rangle|x_0\rangle + (-1)^{H(x_0)+H(x_1)}|1\rangle|x_1\rangle\right)$

Can be done by evaluating H in superposition (in addition to f_λ).

Measuring state in Hadamard basis yields (b, d).

$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness

Intractable to query both $H(x_0)$ and $H(x_1)$ (claw-freeness).

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}} \left(|0\rangle |x_0\rangle + (-1)^{H(x_0)+H(x_1)} |1\rangle |x_1\rangle \right)$
Can be done by evaluating H in superposition (in addition to f_λ).
Measuring state in Hadamard basis yields (b, d).
$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness

Intractable to query both $H(x_0)$ and $H(x_1)$ (claw-freeness).
At least one of $H(x_0), H(x_1)$ will be uniform (ROM).

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}}\left(|0\rangle |x_0\rangle + (-1)^{H(x_0)+H(x_1)} |1\rangle |x_1\rangle\right)$

Can be done by evaluating H in superposition (in addition to f_λ).

Measuring state in Hadamard basis yields (b, d).

$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness

Intractable to query both $H(x_0)$ and $H(x_1)$ (claw-freeness).

At least one of $H(x_0), H(x_1)$ will be uniform (ROM).

$b \oplus d \cdot (x_0 \oplus x_1)$ uncorrelated with $H(x_0) \oplus H(x_1)$.

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}} \left(|0\rangle |x_0\rangle + (-1)^{H(x_0)+H(x_1)} |1\rangle |x_1\rangle \right)$
Can be done by evaluating H in superposition (in addition to f_λ).
Measuring state in Hadamard basis yields (b, d).
$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness $s(\lambda) = \frac{1}{2} + \mathrm{negl}(\lambda)$

Intractable to query both $H(x_0)$ and $H(x_1)$ (claw-freeness).
At least one of $H(x_0), H(x_1)$ will be uniform (ROM).
$b \oplus d \cdot (x_0 \oplus x_1)$ uncorrelated with $H(x_0) \oplus H(x_1)$.

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}} \left(|0\rangle |x_0\rangle + (-1)^{H(x_0)+H(x_1)} |1\rangle |x_1\rangle \right)$
Can be done by evaluating H in superposition (in addition to f_λ).
Measuring state in Hadamard basis yields (b, d).
$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness $s(\lambda) = \frac{1}{2} + \mathrm{negl}(\lambda)$

Intractable to query both $H(x_0)$ and $H(x_1)$ (claw-freeness).
At least one of $H(x_0), H(x_1)$ will be uniform (ROM).
$b \oplus d \cdot (x_0 \oplus x_1)$ uncorrelated with $H(x_0) \oplus H(x_1)$.

Protocol can be parallel repeated to achieve $s(\lambda) = \mathrm{negl}(\lambda)$.

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}}\left(|0\rangle|x_0\rangle + (-1)^{H(x_0)+H(x_1)}|1\rangle|x_1\rangle\right)$

Can be done by evaluating H in superposition (in addition to f_λ).

Measuring state in Hadamard basis yields (b, d).

$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness $s(\lambda) = \frac{1}{2} + \mathrm{negl}(\lambda)$

Intractable to query both $H(x_0)$ and $H(x_1)$ (claw-freeness).

At least one of $H(x_0), H(x_1)$ will be uniform (ROM).

$b \oplus d \cdot (x_0 \oplus x_1)$ uncorrelated with $H(x_0) \oplus H(x_1)$.

Protocol can be parallel repeated to achieve $s(\lambda) = \mathrm{negl}(\lambda)$.

Why not use Fiat-Shamir?

Non-interactive PoQs - BKVV'20

Completeness $c(\lambda) = 1$

Prover prepares state $\frac{1}{\sqrt{2}} \left(|0\rangle |x_0\rangle + (-1)^{H(x_0)+H(x_1)} |1\rangle |x_1\rangle \right)$

Can be done by evaluating H in superposition (in addition to f_λ).

Measuring state in Hadamard basis yields (b, d).

$b = d \cdot (x_0 \oplus x_1) \oplus H(x_0) \oplus H(x_1)$.

Soundness $s(\lambda) = \frac{1}{2} + \mathrm{negl}(\lambda)$

Intractable to query both $H(x_0)$ and $H(x_1)$ (claw-freeness).

At least one of $H(x_0), H(x_1)$ will be uniform (ROM).

$b \oplus d \cdot (x_0 \oplus x_1)$ uncorrelated with $H(x_0) \oplus H(x_1)$.

Protocol can be parallel repeated to achieve $s(\lambda) = \mathrm{negl}(\lambda)$.

Why not use Fiat-Shamir?

BKVV'20 construction only relies on claw-freeness for soundness!

Protocol can use TCFs, rather than STCFs.

Non-interactive PoQs - YZ'22

Surprisingly, in ROM, it's possible to remove the use of TCFs![6]

[6] [Yamakawa, Zhandry, 2022]

Non-interactive PoQs - YZ'22

Surprisingly, in ROM, it's possible to remove the use of TCFs![6]

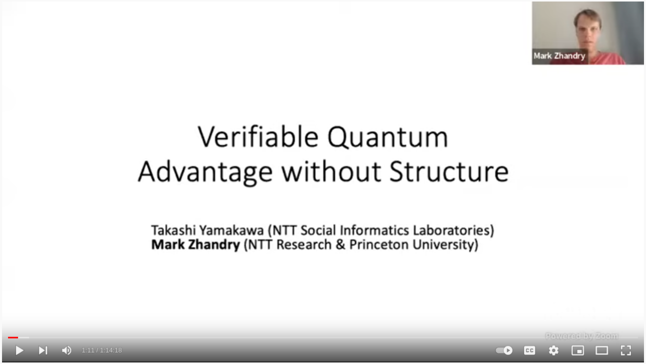
$\mathsf{FBPP}^O \neq \mathsf{FBQP}^O$, where O is a random oracle.

[6] [Yamakawa, Zhandry, 2022]

Non-interactive PoQs - YZ'22

Surprisingly, in ROM, it's possible to remove the use of TCFs![6]

$\mathrm{FBPP}^{O} \neq \mathrm{FBQP}^{O}$, where O is a random oracle.



Verifiable Quantum Advantage Without Structure | Quantum Colloquium

[6] [Yamakawa, Zhandry, 2022]

Non-interactive PoQs - YZ'22

Let Σ be an alphabet of size $2^{\Theta(\lambda)}$,
$C \subseteq \Sigma^n$ ($n = \mathrm{poly}(\lambda)$), be a special linear error-correcting code.
$H : \Sigma \to \{0, 1\}$ is the random oracle.

Non-interactive PoQs - YZ'22

Let Σ be an alphabet of size $2^{\Theta(\lambda)}$,
$C \subseteq \Sigma^n$ ($n = \mathrm{poly}(\lambda)$), be a special linear error-correcting code.
$H : \Sigma \to \{0,1\}$ is the random oracle.

Codeword-finding Problem (CFP)

Given a description of C (parity-check matrix), find a codeword $c = (c_1, c_2, ...c_n) \in C$, such that $H(c_1) = H(c_2) = ...H(c_n) = 0$.

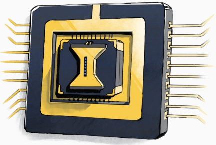
Non-interactive PoQs - YZ'22

Let Σ be an alphabet of size $2^{\Theta(\lambda)}$,
$C \subseteq \Sigma^n$ ($n = \mathrm{poly}(\lambda)$), be a special linear error-correcting code.
$H : \Sigma \to \{0,1\}$ is the random oracle.

Codeword-finding Problem (CFP)

Given a description of C (parity-check matrix), find a codeword $c = (c_1, c_2, \ldots c_n) \in C$, such that $H(c_1) = H(c_2) = \ldots H(c_n) = 0$.

$\longleftarrow (c_1, \ldots, c_n) \in C, H(c_i) = 0$



YZ'22 - Soundness

Consider a poly-time algorithm $\mathcal{A}$ for CFP.

YZ'22 - Soundness

Consider a poly-time algorithm $\mathcal{A}$ for CFP.

For each $1 \leq i \leq n$, denote set of queries to H made by $\mathcal{A}$ as S_i.

$|S_i| = \text{poly}(n)$.

YZ'22 - Soundness

Consider a poly-time algorithm $\mathcal{A}$ for CFP.

For each $1 \leq i \leq n$, denote set of queries to H made by $\mathcal{A}$ as S_i.

$|S_i| = \mathrm{poly}(n)$.

C is *list recoverable*.

There are at most 2^{n^ε} codewords "compatible" with queries from S_i.

YZ'22 - Soundness

Consider a poly-time algorithm $\mathcal{A}$ for CFP.

For each $1 \leq i \leq n$, denote set of queries to H made by $\mathcal{A}$ as S_i.
$|S_i| = \mathrm{poly}(n)$.

C is *list recoverable*.
There are at most $2^{n^{\varepsilon}}$ codewords "compatible" with queries from S_i.

From RO, probability of querying string with all 0 output is 2^{-n}.

YZ'22 - Soundness

Consider a poly-time algorithm $\mathcal{A}$ for CFP.

For each $1 \leq i \leq n$, denote set of queries to H made by $\mathcal{A}$ as S_i.
$|S_i| = \mathrm{poly}(n)$.

C is *list recoverable*.
There are at most 2^{n^ε} codewords "compatible" with queries from S_i.

From RO, probability of querying string with all 0 output is 2^{-n}.

From a union bound, probability $\mathcal{A}$ finds valid codeword is
$2^{n^\varepsilon} \cdot 2^{-n} = \mathrm{negl}(n)$.

YZ'22 - Soundness

Consider a poly-time algorithm $\mathcal{A}$ for CFP.

For each $1 \leq i \leq n$, denote set of queries to H made by $\mathcal{A}$ as S_i.
$|S_i| = \mathrm{poly}(n)$.

C is *list recoverable*.
There are at most $2^{n^{\varepsilon}}$ codewords "compatible" with queries from S_i.

From RO, probability of querying string with all 0 output is 2^{-n}.

From a union bound, probability $\mathcal{A}$ finds valid codeword is
$2^{n^{\varepsilon}} \cdot 2^{-n} = \mathrm{negl}(n)$.

$s(\lambda) = \mathrm{negl}(\lambda)$.

YZ'22 - Completeness

Create the states:

$|\phi\rangle \propto \sum_{c \in C} |c\rangle$ $\quad$ $|\tau\rangle \propto \sum_{w \in \Sigma^n, H(w_1), \ldots, H(w_n) = 0} |w\rangle$

YZ'22 - Completeness

Create the states:

$|\phi\rangle \propto \sum_{c \in C} |c\rangle \qquad |\tau\rangle \propto \sum_{w \in \Sigma^n, H(w_1), \ldots, H(w_n)=0} |w\rangle$

Create the "intersection" (point-wise product) state:

$|\psi\rangle \propto \sum_{c \in C, H(w_1), \ldots, H(w_n)=0} |c\rangle$

YZ'22 - Completeness

Create the states:

$|\phi\rangle \propto \sum_{c \in C} |c\rangle \qquad |\tau\rangle \propto \sum_{w \in \Sigma^n, H(w_1), \ldots, H(w_n)=0} |w\rangle$

Create the "intersection" (point-wise product) state:

$|\psi\rangle \propto \sum_{c \in C, H(w_1), \ldots, H(w_n)=0} |c\rangle$

Measuring $|\psi\rangle$ yields a solution to CFP.

YZ'22 - Completeness

Create the states:

$|\phi\rangle \propto \sum_{c \in C} |c\rangle \qquad |\tau\rangle \propto \sum_{w \in \Sigma^n, H(w_1), \ldots, H(w_n) = 0} |w\rangle$

Create the "intersection" (point-wise product) state:

$|\psi\rangle \propto \sum_{c \in C, H(w_1), \ldots, H(w_n) = 0} |c\rangle$

Measuring $|\psi\rangle$ yields a solution to CFP.

Map $|\phi\rangle, |\tau\rangle$ to Fourier domain (QFT).

YZ'22 - Completeness

Create the states:

$|\phi\rangle \propto \sum_{c \in C} |c\rangle \qquad |\tau\rangle \propto \sum_{w \in \Sigma^n, H(w_1), \ldots, H(w_n)=0} |w\rangle$

Create the "intersection" (point-wise product) state:

$|\psi\rangle \propto \sum_{c \in C, H(w_1), \ldots, H(w_n)=0} |c\rangle$

Measuring $|\psi\rangle$ yields a solution to CFP.

Map $|\phi\rangle, |\tau\rangle$ to Fourier domain (QFT).

Compute convolution in Fourier domain (requires special properties of C).

YZ'22 - Completeness

Create the states:

$|\phi\rangle \propto \sum_{c \in C} |c\rangle$ $\qquad$ $|\tau\rangle \propto \sum_{w \in \Sigma^n, H(w_1), \ldots, H(w_n)=0} |w\rangle$

Create the "intersection" (point-wise product) state:

$|\psi\rangle \propto \sum_{c \in C, H(w_1), \ldots, H(w_n)=0} |c\rangle$

Measuring $|\psi\rangle$ yields a solution to CFP.

Map $|\phi\rangle, |\tau\rangle$ to Fourier domain (QFT).

Compute convolution in Fourier domain (requires special properties of C).

QFT the result yields $|\psi\rangle$ (convolution theorem).

YZ'22 - Completeness

Create the states:

$|\phi\rangle \propto \sum_{c \in C} |c\rangle \qquad |\tau\rangle \propto \sum_{w \in \Sigma^n, H(w_1), \ldots, H(w_n) = 0} |w\rangle$

Create the "intersection" (point-wise product) state:

$|\psi\rangle \propto \sum_{c \in C, H(w_1), \ldots, H(w_n) = 0} |c\rangle$

Measuring $|\psi\rangle$ yields a solution to CFP.

Map $|\phi\rangle, |\tau\rangle$ to Fourier domain (QFT).

Compute convolution in Fourier domain (requires special properties of C).

QFT the result yields $|\psi\rangle$ (convolution theorem).

$c(\lambda) = 1 - \mathrm{negl}(\lambda)$.

Observations about YZ'22

- Non-interactive PoQ in ROM.

Observations about YZ'22

- Non-interactive PoQ in ROM.
- Can be made into a 2-message protocol for non-uniform adversaries.

Observations about YZ'22

- Non-interactive PoQ in ROM.
- Can be made into a 2-message protocol for non-uniform adversaries.
- Quantum strategy involves large circuits (comparable to Shor's algorithm).

Observations about YZ'22

- Non-interactive PoQ in ROM.
- Can be made into a 2-message protocol for non-uniform adversaries.
- Quantum strategy involves large circuits
  (comparable to Shor's algorithm).
- Publicly verifiable (does not involve a trapdoor).

Observations about YZ'22

- Non-interactive PoQ in ROM.
- Can be made into a 2-message protocol for non-uniform adversaries.
- Quantum strategy involves large circuits (comparable to Shor's algorithm).
- Publicly verifiable (does not involve a trapdoor).

Are there other PoQs that do not rely on TCFs?

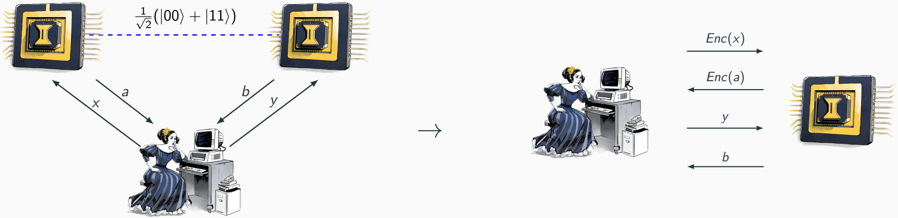
KLVY'22

Non-local games can be turned into proofs of quantumness[7].

[7] [Kalai, Lombardi, Vaikuntanathan, Yang, 2022]

KLVY'22

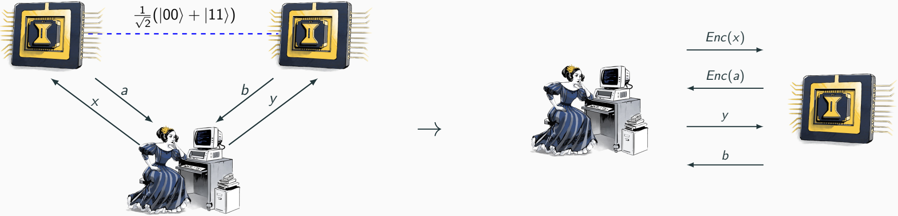
Non-local games can be turned into proofs of quantumness[7].



[7] [Kalai, Lombardi, Vaikuntanathan, Yang, 2022]

KLVY'22

Non-local games can be turned into proofs of quantumness[7].

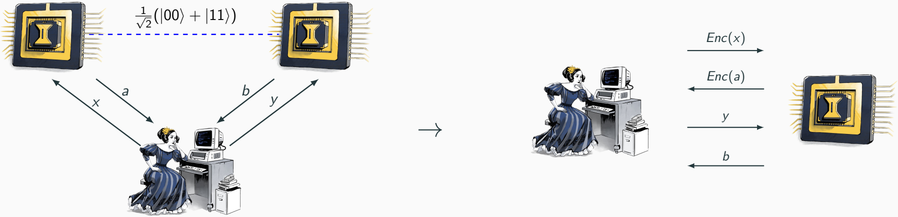


Prover performs quantum strategy of the game *homomorphically*.

[7] [Kalai, Lombardi, Vaikuntanathan, Yang, 2022]

KLVY'22

Non-local games can be turned into proofs of quantumness[7].



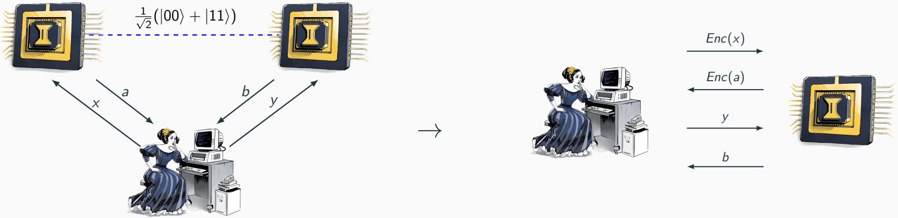
Prover performs quantum strategy of the game *homomorphically*.

Soundness based on security of quantum fully homomorphic encryption.

[7] [Kalai, Lombardi, Vaikuntanathan, Yang, 2022]

KLVY'22

Non-local games can be turned into proofs of quantumness[7].



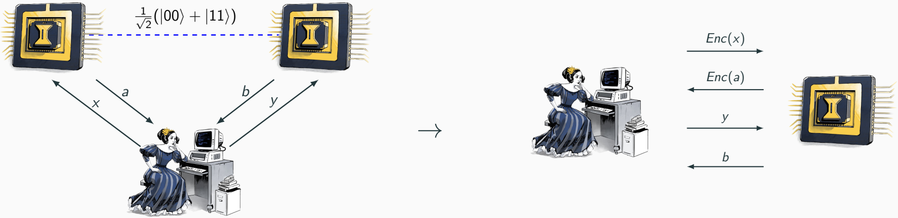
Prover performs quantum strategy of the game *homomorphically*.

Soundness based on security of quantum fully homomorphic encryption.

[7] [Kalai, Lombardi, Vaikuntanathan, Yang, 2022]

KLVY'22

Non-local games can be turned into proofs of quantumness[7].



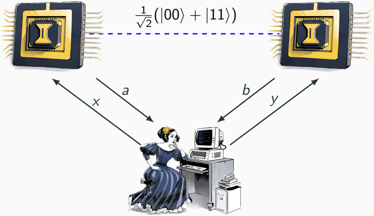
Prover performs quantum strategy of the game *homomorphically*.

Soundness based on security of quantum fully homomorphic encryption.

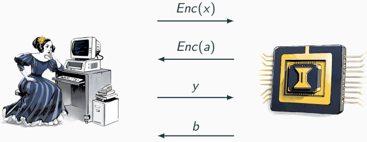
Ironically, QFHE constructions use TCFs[8]. :)

[7] [Kalai, Lombardi, Vaikuntanathan, Yang, 2022]
[8] [Mahadev, 2018], [Brakerski, 2018]

Non-local games can be turned into proofs of quantumness[7].



$\rightarrow$



Prover performs quantum strategy of the game *homomorphically*.

Soundness based on security of quantum fully homomorphic encryption.

Ironically, QFHE constructions use TCFs[8]. :)

If non-local game has quantum completeness c and classical soundness s, PoQ will have $c(\lambda) = c$, $s(\lambda) = s + \mathrm{negl}(\lambda)$.

[7] [Kalai, Lombardi, Vaikuntanathan, Yang, 2022]
[8] [Mahadev, 2018], [Brakerski, 2018]

Observations about KLVY'22

- Not a fixed protocol but a compiler of protocols!

Observations about KLVY'22

- Not a fixed protocol but a compiler of protocols!
- Relates seemingly unrelated sources of quantumness (non-rewinding, non-locality).

Observations about KLVY'22

- Not a fixed protocol but a compiler of protocols!
- Relates seemingly unrelated sources of quantumness (non-rewinding, non-locality).
- QFHE can be based on Ring-LWE (no known adaptive hardcore bit).

Observations about KLVY'22

- Not a fixed protocol but a compiler of protocols!
- Relates seemingly unrelated sources of quantumness (non-rewinding, non-locality).
- QFHE can be based on Ring-LWE (no known adaptive hardcore bit).
- Implementation cost dominated by QFHE (comparable to TCF approaches).

Observations about KLVY'22

- Not a fixed protocol but a compiler of protocols!
- Relates seemingly unrelated sources of quantumness (non-rewinding, non-locality).
- QFHE can be based on Ring-LWE (no known adaptive hardcore bit).
- Implementation cost dominated by QFHE (comparable to TCF approaches).
- QFHE for Cliffords is efficient, but non-local games strategies use T gates as well.

Observations about KLVY'22

- Not a fixed protocol but a compiler of protocols!
- Relates seemingly unrelated sources of quantumness (non-rewinding, non-locality).
- QFHE can be based on Ring-LWE (no known adaptive hardcore bit).
- Implementation cost dominated by QFHE (comparable to TCF approaches).
- QFHE for Cliffords is efficient, but non-local games strategies use T gates as well.
- Efficient if QFHE with $O(1)$ T gates is efficient!

Proofs of quantumness summary

Efficient interactive protocols for classically verifiable quantum advantage.

Proofs of quantumness summary

Efficient interactive protocols for classically verifiable quantum advantage.

- **BCMVV** - 4 messages, STCF (adaptive hardcore bit).
- **KCVY** - 6 messages, TCF, no preimage test.
- **BKVV** - 2 messages, TCF, random oracle.
- **YZ** - 1 (or 2) message(s), random oracle, publicly verifiable.
- **KLVY** - 4 messages, QFHE, general compiler.

Proofs of quantumness summary

Efficient interactive protocols for classically verifiable quantum advantage.

- **BCMVV** - 4 messages, STCF (adaptive hardcore bit).
- **KCVY** - 6 messages, TCF, no preimage test.
- **BKVV** - 2 messages, TCF, random oracle.
- **YZ** - 1 (or 2) message(s), random oracle, publicly verifiable.
- **KLVY** - 4 messages, QFHE, general compiler.

(S)TCF-based constructions can have constant quantum depth prover
(+ log-depth classical computation)

Proofs of quantumness summary

Efficient interactive protocols for classically verifiable quantum advantage.

- **BCMVV** - 4 messages, STCF (adaptive hardcore bit).
- **KCVY** - 6 messages, TCF, no preimage test.
- **BKVV** - 2 messages, TCF, random oracle.
- **YZ** - 1 (or 2) message(s), random oracle, publicly verifiable.
- **KLVY** - 4 messages, QFHE, general compiler.

(S)TCF-based constructions can have constant quantum depth prover (+ log-depth classical computation)

Proof-of-principle demonstration with ion-trap QC show that interaction is in principle feasible in the near-term.

Proofs of quantumness summary

Efficient interactive protocols for classically verifiable quantum advantage.

- **BCMVV** - 4 messages, STCF (adaptive hardcore bit).
- **KCVY** - 6 messages, TCF, no preimage test.
- **BKVV** - 2 messages, TCF, random oracle.
- **YZ** - 1 (or 2) message(s), random oracle, publicly verifiable.
- **KLVY** - 4 messages, QFHE, general compiler.

(S)TCF-based constructions can have constant quantum depth prover (+ log-depth classical computation)

Proof-of-principle demonstration with ion-trap QC show that interaction is in principle feasible in the near-term.

Current estimates for quantum advantage demonstration: $\sim 1000 - 2000$ qubits and 10^5 layers of depth.

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?

[9] Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?
4. Do TCFs that are not STCFs exist or is every TCF also a STCF?

[9] Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?
4. Do TCFs that are not STCFs exist or is every TCF also a STCF?
5. Are there more efficient quantum constructions with potentially higher classical overhead?

[9] Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?
4. Do TCFs that are not STCFs exist or is every TCF also a STCF?
5. Are there more efficient quantum constructions with potentially higher classical overhead?
6. Other than YZ'22, can PoQs be made publicly verifiable?

[9]Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?
4. Do TCFs that are not STCFs exist or is every TCF also a STCF?
5. Are there more efficient quantum constructions with potentially higher classical overhead?
6. Other than YZ'22, can PoQs be made publicly verifiable?
7. Better protocols (non-interactivity, public verifiability, efficiency) from stronger crypto assumptions (post-quantum iO)?

[9] Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?
4. Do TCFs that are not STCFs exist or is every TCF also a STCF?
5. Are there more efficient quantum constructions with potentially higher classical overhead?
6. Other than YZ'22, can PoQs be made publicly verifiable?
7. Better protocols (non-interactivity, public verifiability, efficiency) from stronger crypto assumptions (post-quantum iO)?
8. Finer grained proofs of quantumness (proofs of quantum depth[10] [11]).

[9] Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

[10] [Chia, Hung, 2022]

[11] Coming soon... [Arora, Coladangelo, Coudron, Gheorghiu, Singh, Waldner]

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?
4. Do TCFs that are not STCFs exist or is every TCF also a STCF?
5. Are there more efficient quantum constructions with potentially higher classical overhead?
6. Other than YZ'22, can PoQs be made publicly verifiable?
7. Better protocols (non-interactivity, public verifiability, efficiency) from stronger crypto assumptions (post-quantum iO)?
8. Finer grained proofs of quantumness (proofs of quantum depth[10] [11]).
9. Can we combine PoQs with other tests of quantum advantage (sampling-based approaches)?

[9] Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

[10] [Chia, Hung, 2022]

[11] Coming soon... [Arora, Coladangelo, Coudron, Gheorghiu, Singh, Waldner]

Open problems

1. More efficient constructions (fewer gates and qubits, short depth).
2. Elliptic-curve-based TCF (more efficient than Shor's algorithm)?
3. Can BCMVV preimage test be removed[9]?
4. Do TCFs that are not STCFs exist or is every TCF also a STCF?
5. Are there more efficient quantum constructions with potentially higher classical overhead?
6. Other than YZ'22, can PoQs be made publicly verifiable?
7. Better protocols (non-interactivity, public verifiability, efficiency) from stronger crypto assumptions (post-quantum iO)?
8. Finer grained proofs of quantumness (proofs of quantum depth[10] [11]).
9. Can we combine PoQs with other tests of quantum advantage (sampling-based approaches)?

Thanks!

[9] Maybe, with non-falsifiable assumptions (based on discussions with Brakerski, Mahadev, Metger, Vaikuntanathan, Wright).

[10] [Chia, Hung, 2022]

[11] Coming soon... [Arora, Coladangelo, Coudron, Gheorghiu, Singh, Waldner]