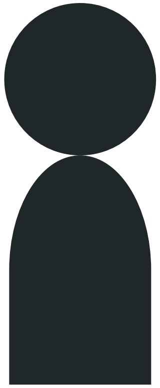
Classical Verification of Quantum Depth

Nai-Hui Chia (Rice University)

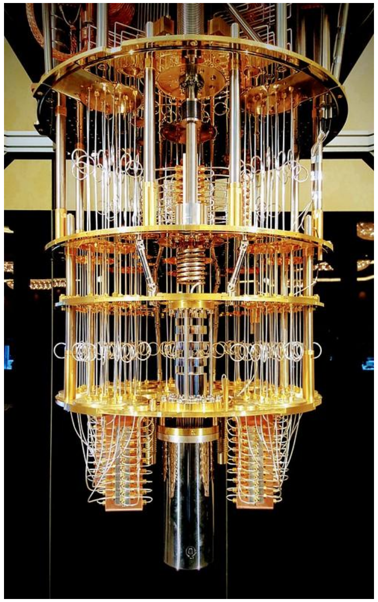
Based on joint work with Shih-Hang Hung

Certifying Quantum Resources

Can classical clients verify that a remote server has claimed quantum resources?



Classical client

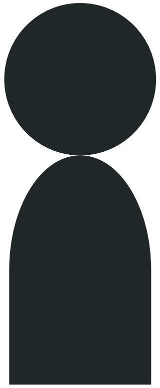


Remote server

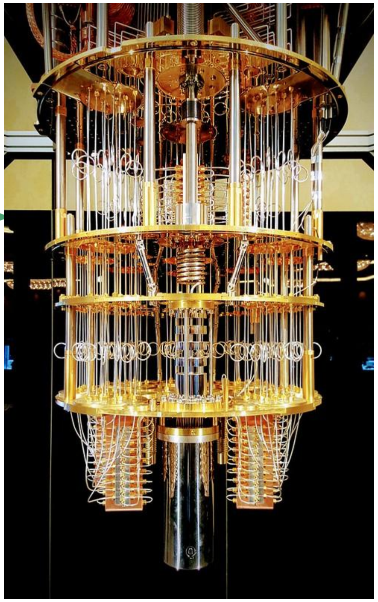
Certifying Quantum Resources

Can classical clients verify that a remote server has claimed quantum resources?

Do you have 500 qubits and 100 depths?



Classical client



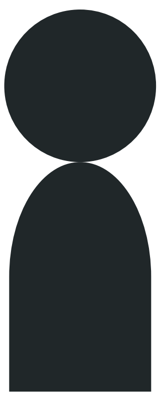
Remote server

Certifying Quantum Resources

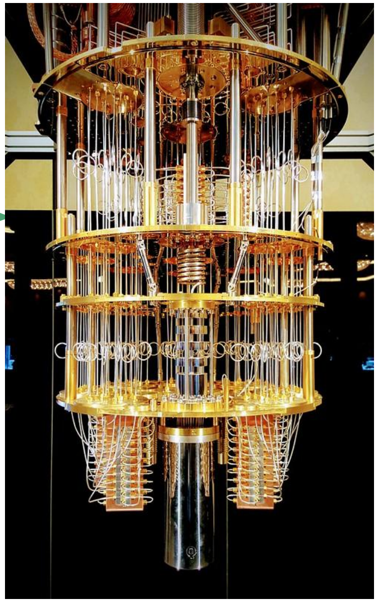
Can classical clients verify that a remote server has claimed quantum resources?

Do you have 500 qubits and 100 depths?

Yes!



Classical client



Remote server

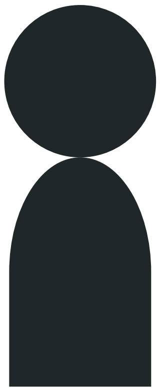
Certifying Quantum Resources

Can classical clients verify that a remote server has claimed quantum resources?

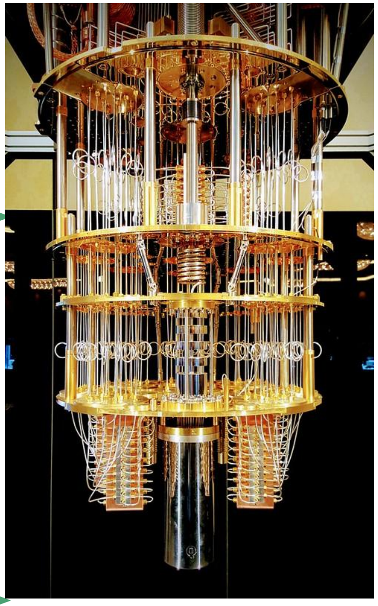
Do you have 500 qubits and 100 depths?

Yes!

Okay... I will pay you for the quantum computer...



Classical client



Remote server

Certifying C

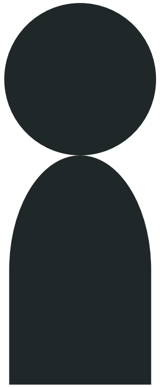
Actually, I only have 50 qubits and 10 depths. But you never know -- I can use some supercomputer to complete you task!

Can classical clients verify ... erver has claimed quantum resources?

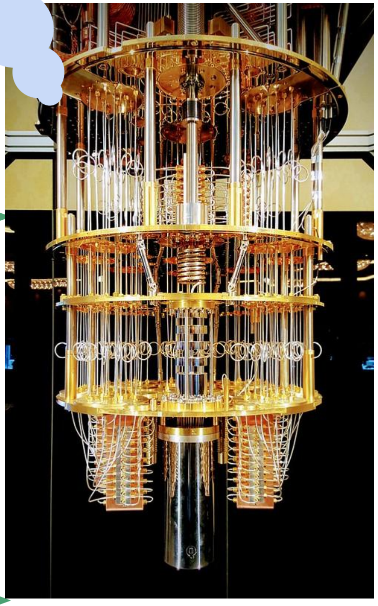
Do you have 500 qubits and 100 depths?

Yes!

Okay... I will pay you for the quantum computer...



Classical client



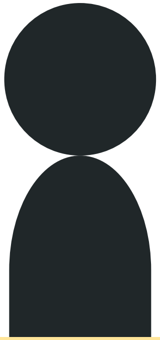
Remote server

Certifying C

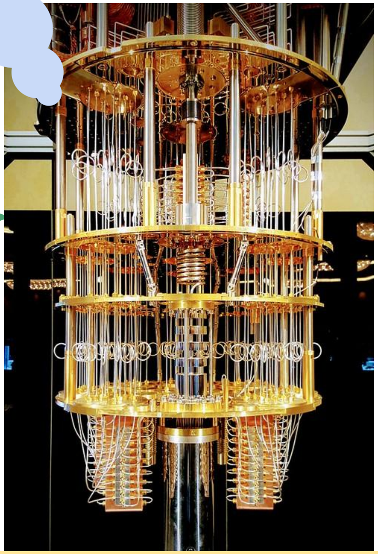
Actually, I only have 50 qubits and 10 depths. But you never know -- I can use some supercomputer to complete you task!

Can classical clients verify th erver has claimed quantum resources?

Do you have 500 qubits and 100 depths?



Yes!



Okay... I will pay you for the quantum computer...

We are especially interested in protocols for certifying **quantum depth**

Why do we care about quantum depth?

Near-term QC:

- > 100 qubits
- Noisy gates
  (not fault-tolerant)
  + short coherence time

Why do we care about quantum depth?

Near-term QC:

- > 100 qubits
- Noisy gates
  (not fault-tolerant)
  + short coherence time
  => Small circuit depth.
- Google's supremacy experiment:
  - 53 qubits + 20 depth
    => 99.8% noise

Why do we care about quantum depth?

Near-term QC:

- > 100 qubits
- Noisy gates
  (not fault-tolerant)
  + short coherence time
  => Small circuit depth
- Google's supremacy experiment:
  - 53 qubits + 20 depth
    => 99.8% noise

General QC:

- Many qubits
- Small noise (fault-tolerant)
  + long coherence time
  => Poly(n) circuit depth
- Can do all poly-time (quantum) algorithms

Building machine with large quantum depth is challenging

Why do we care about quantum depth?

Near-term QC:

- > 100 qubits
- Noisy gates
  (not fault-tolerant)
  + short coherence time
  => Small circuit depth
- Google's supremacy experiment:
  - 53 qubits + 20 depth
    => 99.8% noise

General QC:

- Many qubits
- Small noise (fault-tolerant)
  + long coherence time
  => Poly(n) circuit depth
- Can do all poly-time (quantum) algorithms

Building machine with large quantum depth is challenging

Goal: A method to check quantum depth of a remote server

Classical Verification of Quantum Depth (CVQD)

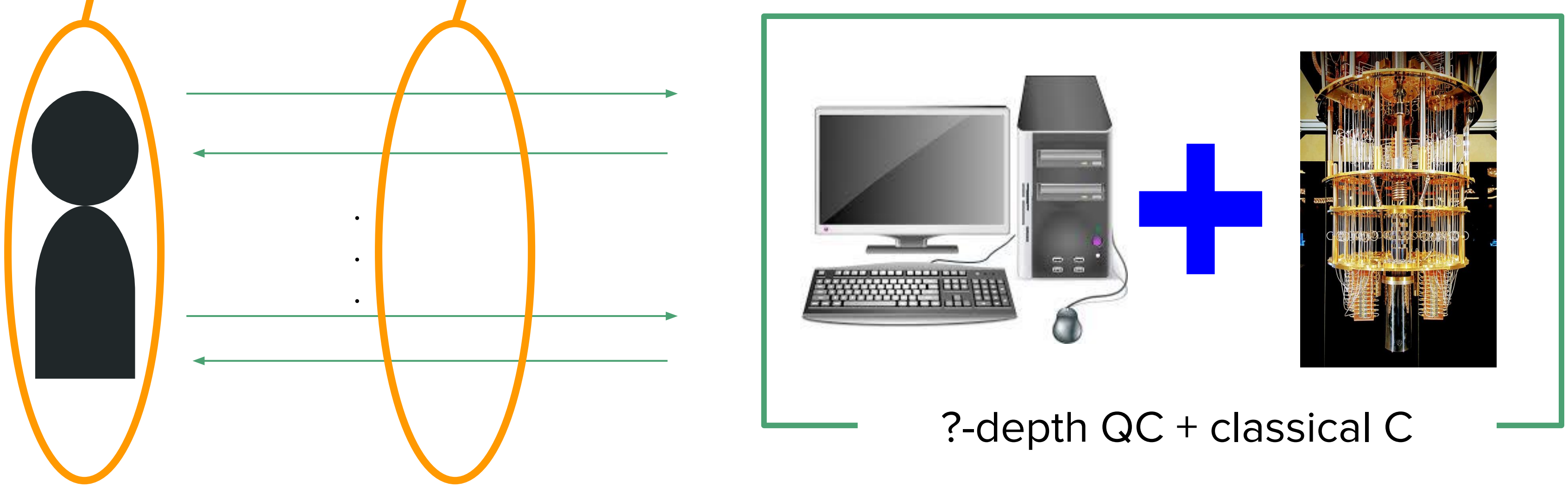
A protocol which lets a classical verifier check the quantum depth of a remote server

Classical Verification of Quantum Depth (CVQD)

A protocol which lets a classical verifier check the quantum depth of a remote server

Three requirements for CVQD

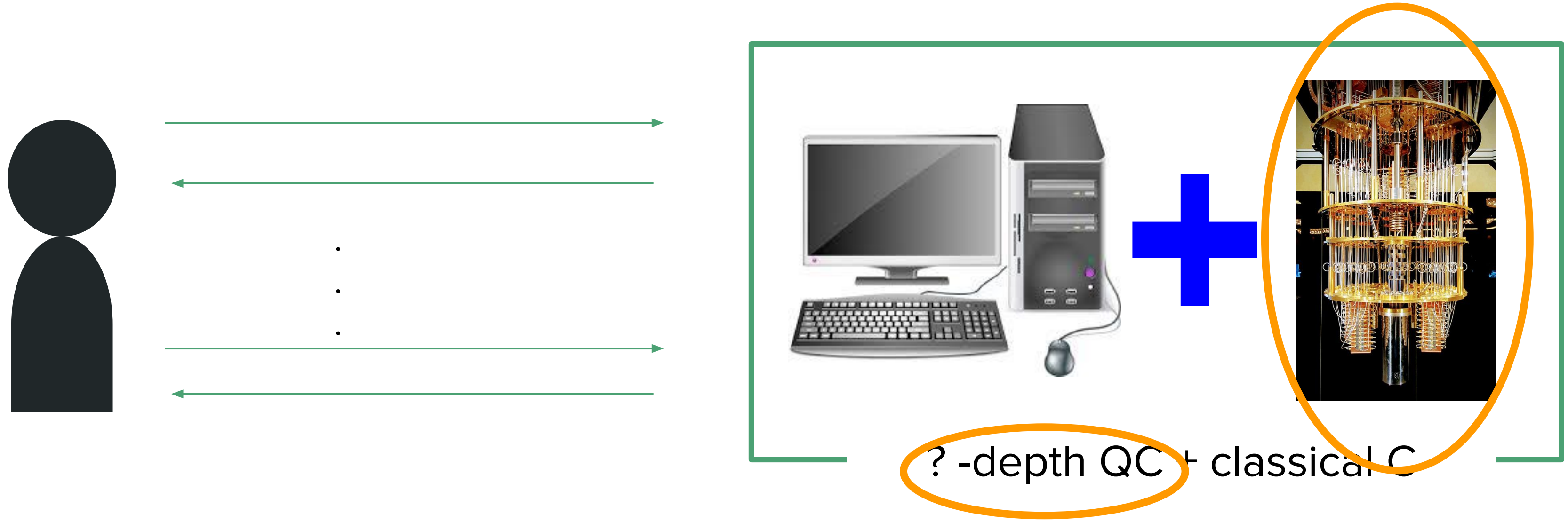
1. **Classical** verifier and **classical** protocol
2.
3.



Classical Verification of Quantum Depth (CVQD)

Three requirements for CVQD

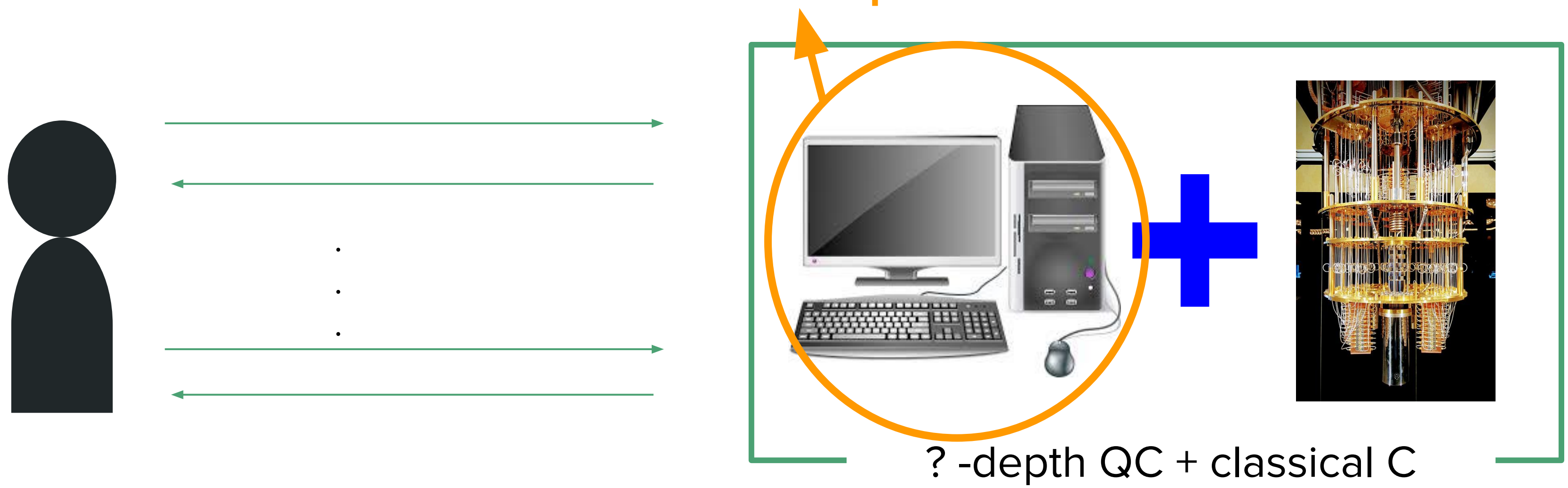
1. Classical verifier and classical protocol
2. Recognize **server's quantum depth**
   a. Server's quantum depth > d => Verifier accepts
   b. Server's quantum depth ≤ d => Verifier rejects
3.



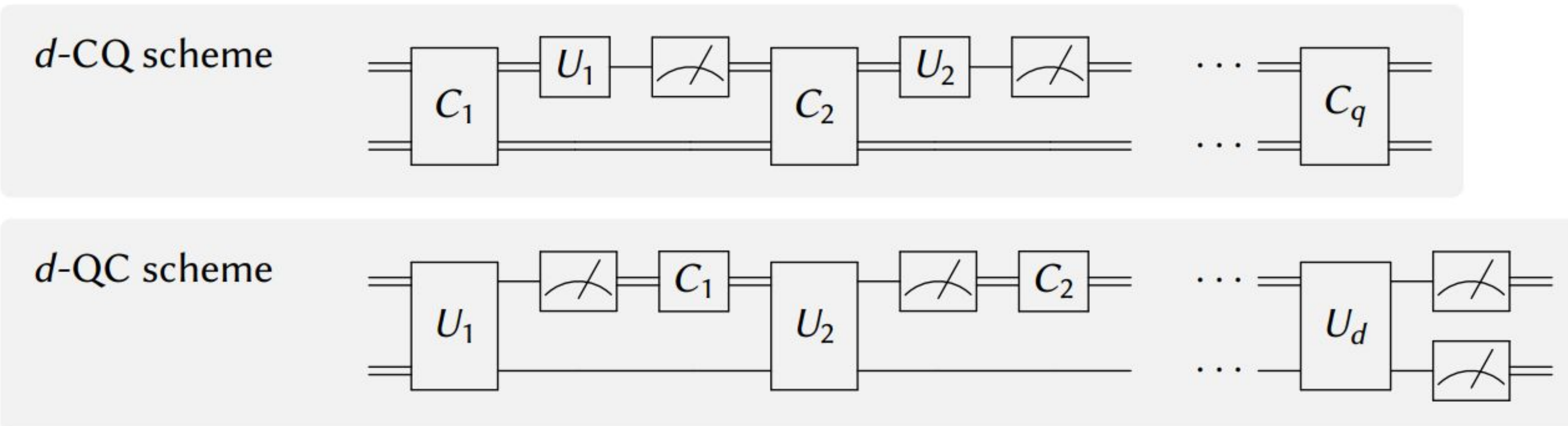
Classical Verification of Quantum Depth (CVQD)

Three requirements for CVQD

1. Classical verifier and classical protocol
2. Recognize server's quantum depth
   a. Server's quantum depth > d => Verifier accepts
   b. Server's quantum depth ≤ d => Verifier rejects
3. The remote server can **use classical computation** to cheat!



Two Models of Hybrid Quantum-Classical Comp.



- *d*-CQ scheme: classical computer can access a *d*-depth quantum circuit
- *d*-QC scheme: *d*-depth quantum circuits can access classical computer after each layer of gates.

First Attempt for CVQD protocols

First Attempt for CVQD Protocols

Theorem [**C**CL20, CH22, HG22]: ∃ a problem (called d-SSP) such that

- **[(d+3)**-depth QC + Classical C] solves the problem
- **[d**-depth QC+Classical C] cannot solve the problem

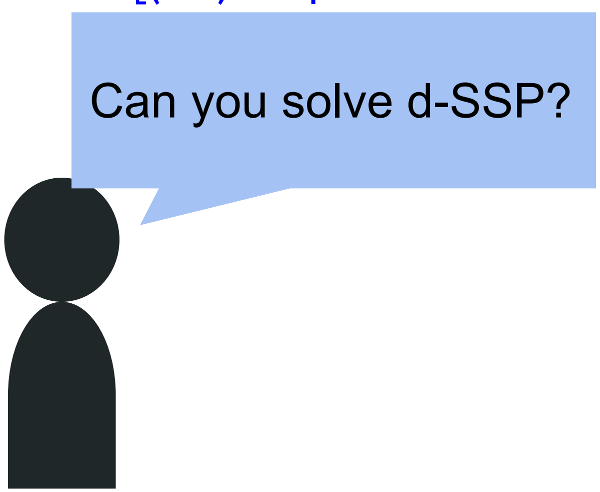
First Attempt for CVQD Protocols

Theorem [**C**CL20, CH22, HG22]: ∃ a problem (called d-SSP) such that

- **[(d+3)**-depth QC + Classical C] solves the problem
- **[d**-depth QC+Classical C] cannot solve the problem

Idea: The verifier asks the remote server to solve the d-SSP

- [(d+3)-depth QC + Classical C] can solve the problem
- [(≤d)-depth QC + Classical C] cannot solve the problem



First Attempt for CVQD Protocols

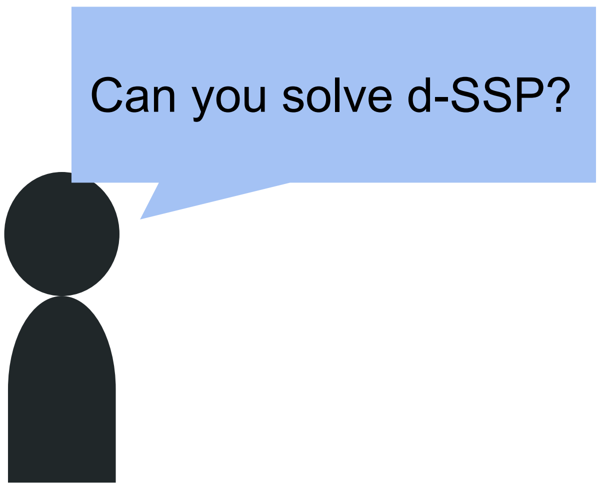
Theorem [**C**CL20, CH22, HG22]: ∃ a problem (called d-SSP) such that

- **[(d+3)**-depth QC + Classical C] solves the problem
- **[d**-depth QC+Classical C] cannot solve the problem

Oops!

Idea: The verifier asks the remote server to solve the d-

- [(d+3)-depth QC + Classical C] can solve the prob
- [(≤d)-depth QC + Classical C] cannot solve the proble



? -depth QC + classical C

This idea satisfies all requirements of CVQD?

First Attempt for CVQD Protocols

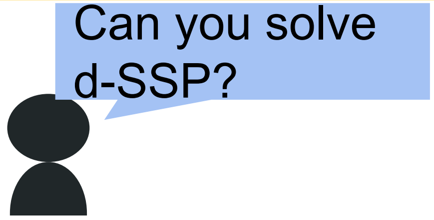
Idea: The verifier asks the remote server to solve the d-SSP
- [(d+3)-depth QC + Classical C] can solve the problem
- [(≤d)-depth QC + Classical C] cannot solve the problem

This idea satisfies all requirements of CVQD?

First Attempt for CVQD Protocols

Idea: The verifier asks the remote server to solve the d-SSP

- [(d+3)-depth QC + Classical C] can solve the problem
- [(≤d)-depth QC + Classical C] cannot solve the problem



Verifier

Server

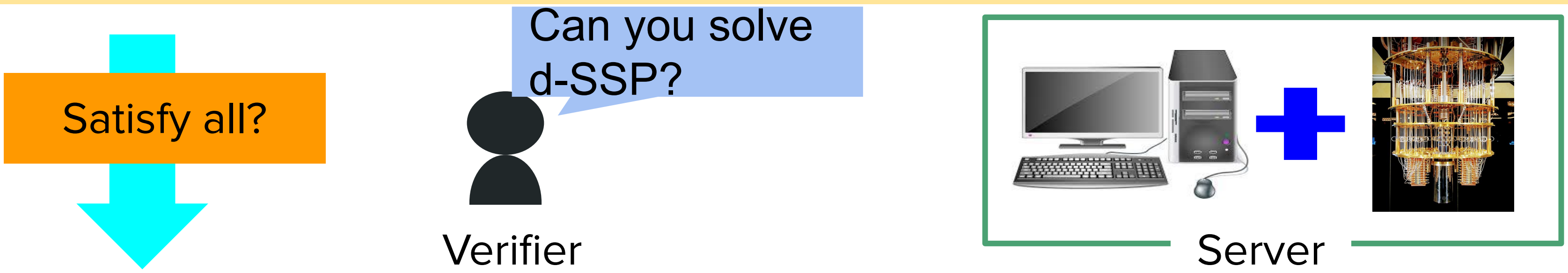
Three requirements for CVQD

1. Classical verifier and classical protocol
2. Recognize server’s quantum depth
    a. Server’s quantum depth > d’ => Verifier accepts
    b. Server’s quantum depth ≤ d => Verifier rejects
3. The remote server can use classical computation to cheat!

This idea satisfies all requirements of CVQD?

First Attempt for CVQD Protocols

Idea: The verifier asks the remote server to solve the d-SSP
- [(d+3)-depth QC + Classical C] can solve the problem
- [(≤d)-depth QC + Classical C] cannot solve the problem



Three requirements for CVQD

1. Classical verifier and classical protocol
2. Recognize server’s quantum depth
    a. Server’s quantum depth > d’ => Verifier accepts
    b. Server’s quantum depth ≤ d => Verifier rejects
3. The remote server can use classical computation to cheat!

This idea satisfies all requirements of CVQD?

First Attempt for CVQD Protocols

Idea: The verifier asks the remote server to solve the d-SSP
- [(d+3)-depth QC + Classical C] can solve the problem
- [(≤d)-depth QC + Classical C] cannot solve the problem

Satisfy all?

Separate d+3 quantum depth from d.

Three requirements for CVQD

1. Classical verifier and classical protocol
2. Recognize server's quantum depth
   a. Server's quantum depth > d' => Verifier accepts
   b. Server's quantum depth ≤ d => Verifier rejects
3. The remote server can use classical computation to cheat!

This idea satisfies all requirements of CVQD?

First Attempt for CVQD Protocols

Idea: The verifier asks the remote server to solve the d-SSP

- [(d+3)-depth QC + Classical C] can solve the problem
- [(≤d)-depth QC + Classical C] cannot solve the problem

Satisfy all?

Server without sufficient quantum depth cannot convince the verifier even using classical computer

Three requirements for CVQD

1. Classical verifier and classical protocol
2. Recognize server's quantum depth
   a. Server's quantum depth > d' => Verifier accepts
   b. Server's quantum depth ≤ d => Verifier rejects
3. The remote server can use classical computation to cheat!

This idea satisfies all requirements of CVQD?

First Attempt for CVQD Protocols

Idea: The verifier asks the remote server to solve the d-SSP
- [(d+3)-depth QC + Classical C] can solve the problem
- [(≤d)-depth QC + Classical C] cannot solve the problem

Satisfy all?

Are the verifier and the protocol classical?

Three requirements for CVQD

1. Classical verifier and classical protocol
2. Recognize server's quantum depth
   a. Server's quantum depth > d' => Verifier accepts
   b. Server's quantum depth ≤ d => Verifier rejects
3. The remote server can use classical computation to cheat!

Are the verifier and the protocol classical?

Are the verifier and the protocol classical?

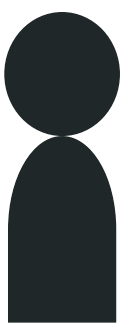
***d*-SSP:**
Input: quantum oracle **F= {f_0,..., f_d}** encoding a **Simon's problem**,
Goal: solved the Simon's problem

Are the verifier and the protocol classical?

***d*-SSP:**
Input: quantum oracle $\mathbf{F} \coloneqq \{f_0,\dots, f_d\}$ encoding a **Simon's problem**,
Goal: solved the Simon's problem

1. The verifier implements the **quantum** oracle **F**

$\mathbf{F} \coloneqq \{f_0,\dots, f_d\}$

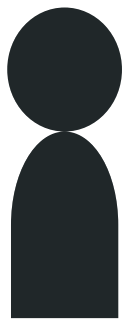


Are the verifier and the protocol classical?

***d*-SSP:**
Input: quantum oracle $\mathbf{F} \coloneqq \{\mathbf{f}_0,\dots,\mathbf{f}_d\}$ encoding a **Simon's problem**,
Goal: solved the Simon's problem

1. The verifier implements the **quantum** oracle **F**
2. The server runs the algorithm

$\mathbf{F} \coloneqq \{\mathbf{f}_0,\dots,\mathbf{f}_d\}$



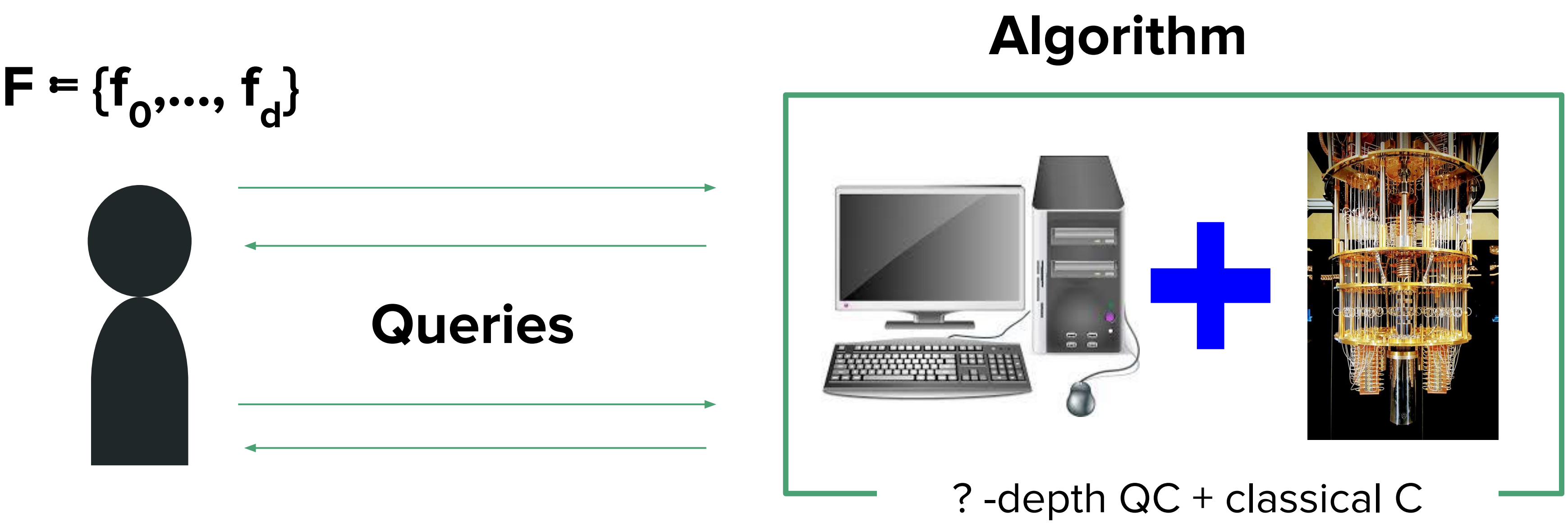
Algorithm

? -depth QC + classical C

Are the verifier and the protocol classical?

***d*-SSP:**
Input: quantum oracle $\mathbf{F} \Leftarrow \{f_0, \dots, f_d\}$ encoding a **Simon's problem**,
Goal: solved the Simon's problem

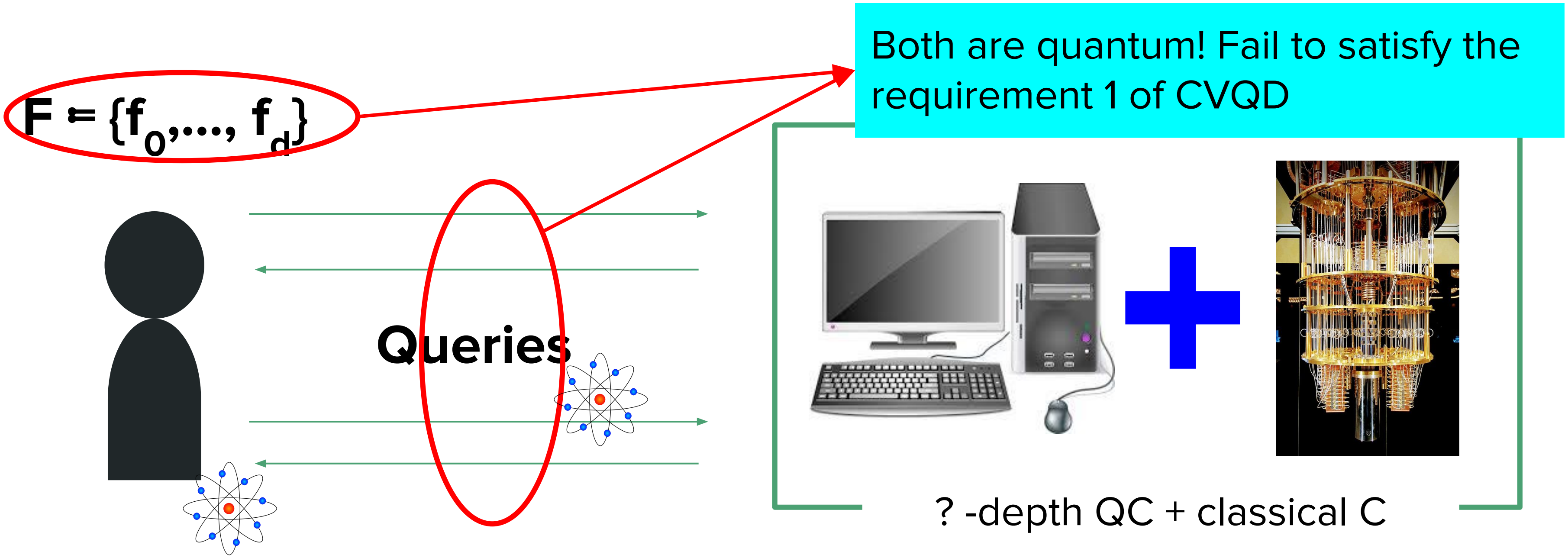
1. The verifier implements the **quantum** oracle **F**
2. The server runs the algorithm
3. Exchange **quantum messages** for quantum queries



Are the verifier and the protocol classical?

***d*-SSP:**
Input: quantum oracle $\mathbf{F} \leftarrow \{\mathbf{f}_0, \dots, \mathbf{f}_d\}$ encoding a **Simon's problem**,
Goal: solved the Simon's problem

1. The verifier implements the **quantum** oracle **F**
2. The server runs the algorithm
3. Exchange **quantum** **messages** for quantum queries



Our CVQD Protocols

Two CVQD protocols:

Theorem 1 [Chia-Hung 22]: ∃ two CVQD protocols where

- **Protocol A** distinguishes **d-depth quantum circuit** from **d+c*** for any d

Two CVQD protocols:

Theorem 1 [Chia-Hung 22]: ∃ two CVQD protocols where

- **Protocol A** distinguishes **d-depth quantum circuit** from **d+c*** for any d

Protocol A

1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
   a. **(Completeness)** Server's quantum depth > **d+c*** => Verifier accepts
   b. **(Soundness)** Server's quantum depth ≤ **d** => Verifier rejects
3. The remote server can **use classical computation** to cheat!

Two CVQD protocols:

Theorem 1 [Chia-Hung 22]: ∃ two CVQD protocols where

- **Protocol A** distinguishes **d-depth quantum circuit** from **d+c*** for any d

Protocol A

1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
   a. **(Completeness)** Server's quantum depth > **d+c*** => Verifier accepts
   b. **(Soundness)** Server's quantum depth ≤ **d** => Verifier rejects
3. The remote server can **use classical computation** to cheat!

Remarks of Protocol A

- The separation **(d v.s. d+c*) is not optimal**
- Honest server **requires (>c*)-depth** to implement the protocol
- Other improvements that we will discuss later

Two CVQD protocols:

Theorem 1 [Chia-Hung 22]: ∃ two CVQD protocols where

- **Protocol A** distinguishes **d-depth quantum circuit** from **d+c*** for any d
- **Protocol B, a two-prover protocol,** distinguishes **d-depth quantum circuit** from **d+3** for any d

Two CVQD protocols:

Theorem 1 [Chia-Hung 22]: ∃ two CVQD protocols where

- **Protocol A** distinguishes **d-depth quantum circuit** from **d+c*** for any d
- **Protocol B, a two-prover protocol,** distinguishes **d-depth quantum circuit** from **d+3** for any d

Protocol B

1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
    a. **(Completeness)** Server's quantum depth > **d+3** => Verifier accepts
    b. **(Soundness)** Server's quantum depth ≤ **d** => Verifier rejects
3. The remote server can **use classical computation** to cheat!

Two CVQD protocols:

Theorem 1 [Chia-Hung 22]: ∃ two CVQD protocols where

- **Protocol A** distinguishes **d-depth quantum circuit** from **d+c*** for any d
- **Protocol B, a two-prover protocol,** distinguishes **d-depth quantum circuit** from **d+3** for any d

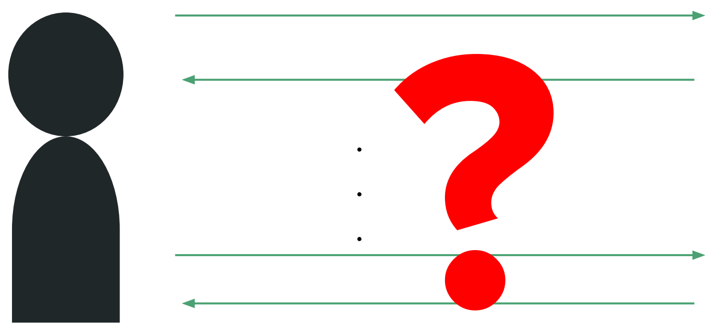
Protocol B

1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
   a. **(Completeness)** Server's quantum depth > **d+3** => Verifier accepts
   b. **(Soundness)** Server's quantum depth ≤ **d** => Verifier rejects
3. The remote server can **use classical computation** to cheat!

Remarks of Protocol B

Better separation (d versus d+3), but protocol B **requires another dishonest quantum prover** to help.

What is our protocol?



? -depth QC + classical C

Protocol A

Idea:

Proof of Quantumness Protocol + “Pointer chasing”

Proof of Quantumness (PoQ)

BCMVV’s proof of quantumness protocol (PoQ)

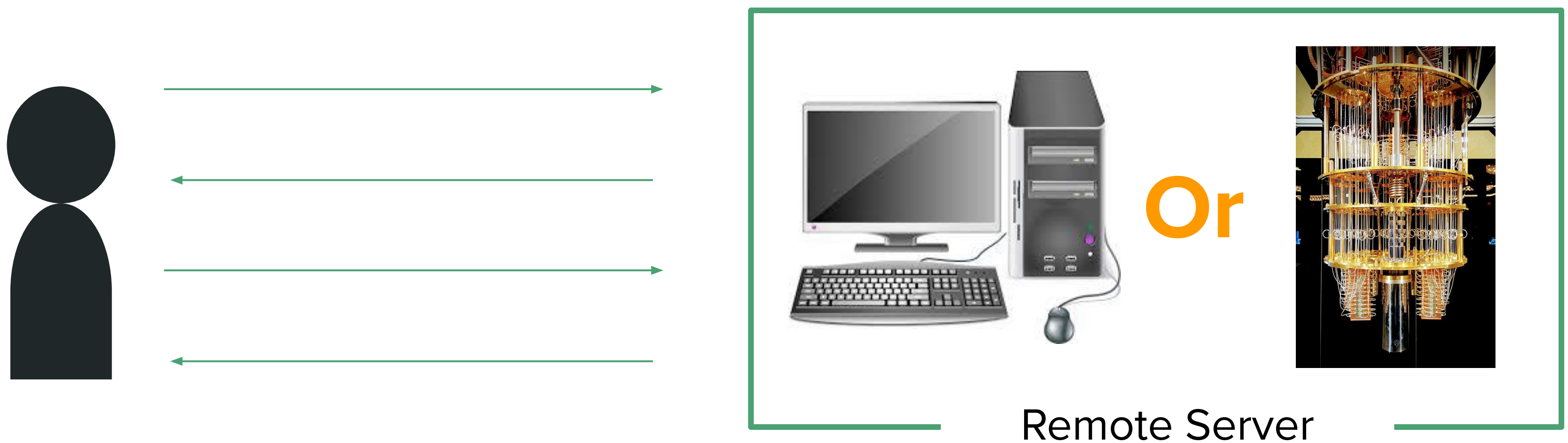
- Goal: Let a classical client to distinguish the following two cases
  a. A remote server has specific quantum power
  b. A remote server is purely classical

[Brakerski, Christiano, Mahadev, Vazirani, Vidick ’18]

Proof of Quantumness (PoQ)

BCMVV's proof of quantumness protocol (PoQ)

- Goal: Let a classical client to distinguish the following two cases
  a. A remote server has specific quantum power
  b. A remote server is purely classical

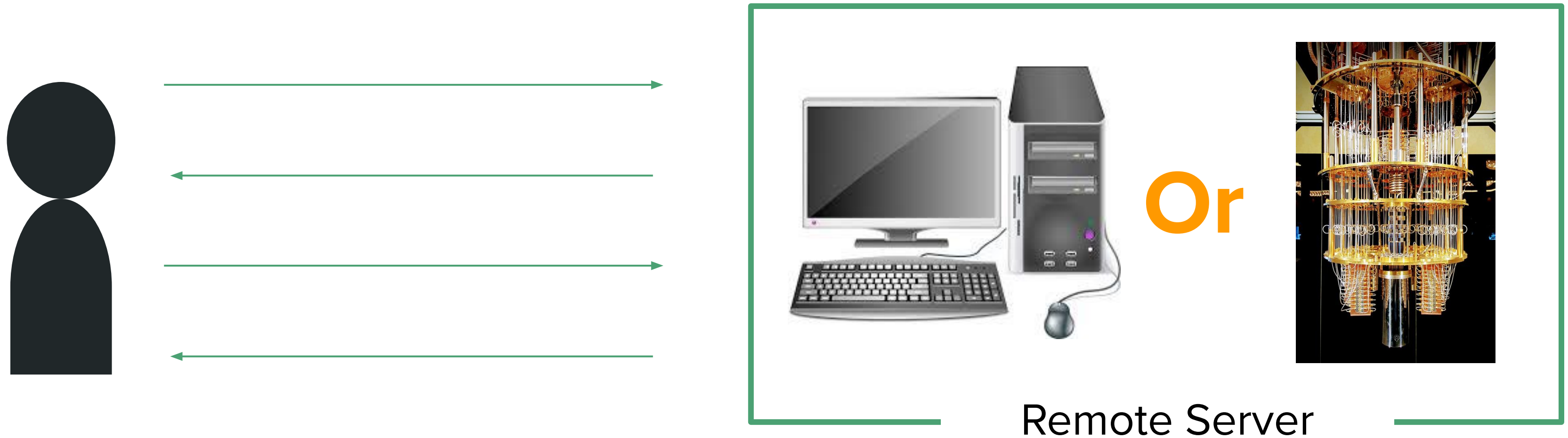


[Brakerski, Christiano, Mahadev, Vazirani, Vidick '18]

Proof of Quantumness (PoQ)

BCMVV's proof of quantumness protocol (PoQ) from Adaptive Hardcore Bit (AHB)

- Goal: Let a classical client to distinguish the following two cases
  a. A remote server has specific quantum power
  b. A remote server is purely classical
- AHB: From LWE, ∃ distribution over 2-to-1 functions f such that
  a. No efficient Q. alg. can output (y, x, e) s.t. f(x) = y and $(x_0+x_1) \cdot e = 0$ w.p. > 1/2
  b. After revealing y, (y, x) or (y, e) can be efficiently computed (by Q. alg.)

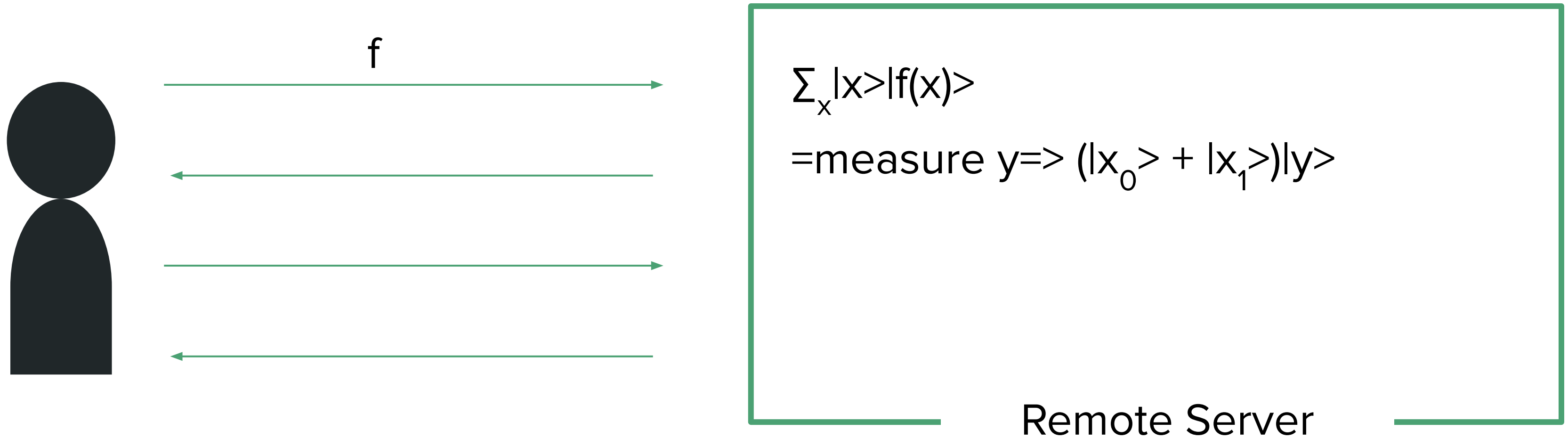


[Brakerski, Christiano, Mahadev, Vazirani, Vidick '18]

Proof of Quantumness (PoQ)

BCMVV's proof of quantumness protocol (PoQ) from Adaptive Hardcore Bit (AHB)

- Goal: Let a classical client to distinguish the following two cases
    a. A remote server has specific quantum power
    b. A remote server is purely classical
- AHB: From LWE, ∃ distribution over 2-to-1 functions f such that
    a. No efficient Q. alg. can output (y, x, e) s.t. $f(x) = y$ and $(x_0+x_1) \cdot e = 0$ w.p. > 1/2
    b. After revealing y, (y, x) or (y, e) can be efficiently computed (by Q. alg.)

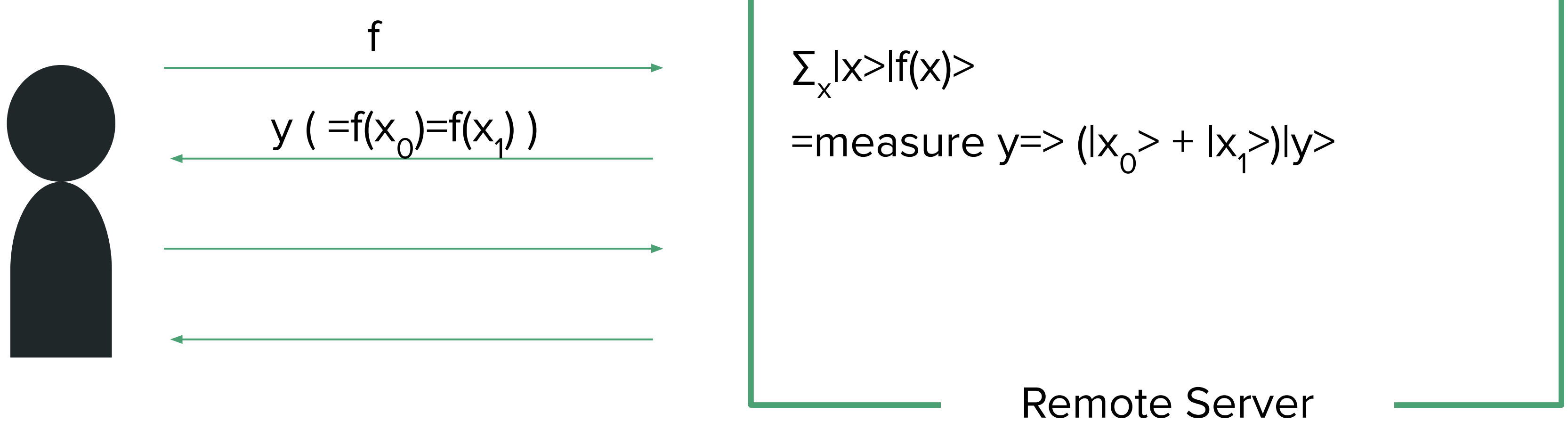


[Brakerski, Christiano, Mahadev, Vazirani, Vidick '18]

Proof of Quantumness (PoQ)

BCMVV's proof of quantumness protocol (PoQ) from Adaptive Hardcore Bit (AHB)

- Goal: Let a classical client to distinguish the following two cases
  a. A remote server has specific quantum power
  b. A remote server is purely classical
- AHB: From LWE, ∃ distribution over 2-to-1 functions f such that
  a. No efficient Q. alg. can output (y, x, e) s.t. $f(x) = y$ and $(x_0+x_1) \cdot e = 0$ w.p. > 1/2
  b. After revealing y, (y, x) or (y, e) can be efficiently computed (by Q. alg.)

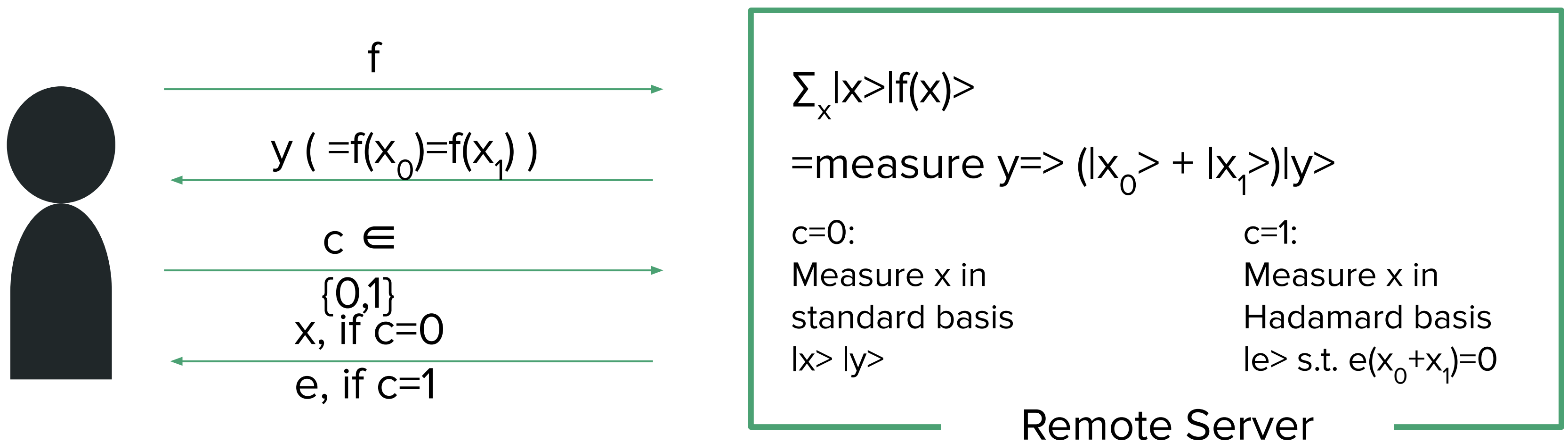


[Brakerski, Christiano, Mahadev, Vazirani, Vidick '18]

Proof of Quantumness (PoQ)

BCMVV's proof of quantumness protocol (PoQ) from Adaptive Hardcore Bit (AHB)

- Goal: Let a classical client to distinguish the following two cases
  a. A remote server has specific quantum power
  b. A remote server is purely classical
- AHB: From LWE, ∃ distribution over 2-to-1 functions f such that
  a. No efficient Q. alg. can output (y, x, e) s.t. $f(x) = y$ and $(x_0+x_1) \cdot e = 0$ w.p. > 1/2
  b. After revealing y, (y, x) or (y, e) can be efficiently computed (by Q. alg.)

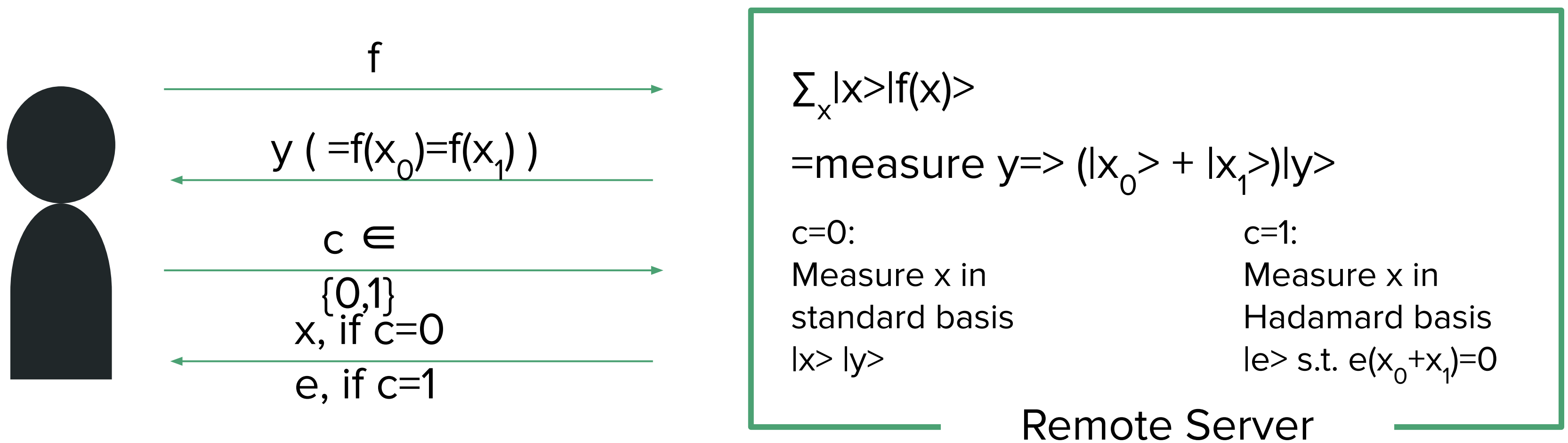


[Brakerski, Christiano, Mahadev, Vazirani, Vidick '18]

- A classical server who can win the game can break AHB by rewinding!
- [Hirahara-Le Gall '21] and [Liu-Gheorghiu '21] showed that f can be evaluated in depth-O(1) (i.e., c*) ⇒ separates depth-O(1) from depth-0 (i.e., classical) device.

BCMVV's proof of quantumness protocol (PoQ) from Adaptive Hardcore Bit (AHB)

- Goal: Let a classical client to distinguish the following two cases
    a. A remote server has specific quantum power
    b. A remote server is purely classical
- AHB: From LWE, ∃ distribution over 2-to-1 functions f such that
    a. No efficient Q. alg. can output (y, x, e) s.t. $f(x) = y$ and $(x_0+x_1) \cdot e = 0$ w.p. > 1/2
    b. After revealing y, (y, x) or (y, e) can be efficiently computed (by Q. alg.)



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

PoQ + “Pointer Chasing” = CVQD!

- Goal: Distinguish a depth-$(d + c^*)$ from a depth-d QC scheme

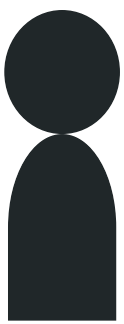
PoQ + “Pointer Chasing” = CVQD!

- Goal: Distinguish a depth-$(d + c^*)$ from a depth-d QC scheme
- **Pointer chasing:** a sequence of questions where the next question is determined by the last answer.

PoQ + “Pointer Chasing” = CVQD!

- Goal: Distinguish a depth-(d + c*) from a depth-d QC scheme
- Pointer chasing: a sequence of questions where the next question is determined by the last answer.

$f_1, f_2, \ldots, f_d$



$\Sigma_x|x\rangle|f_1(x)\rangle, \Sigma_x|x\rangle|f_2(x)\rangle, \ldots, \Sigma_x|x\rangle|f_d(x)\rangle$

$\Rightarrow (|x_0^1\rangle + |x_1^1\rangle)|y_1\rangle, \ldots, (|x_0^d\rangle + |x_1^d\rangle)|y_d\rangle$

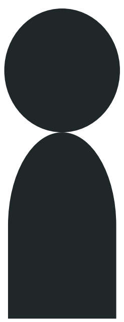
Remote Server

PoQ + “Pointer Chasing” = CVQD!

- Goal: Distinguish a depth-(d + c*) from a depth-d QC scheme
- Pointer chasing: a sequence of questions where the next question is determined by the last answer.

$f_1,f_2,\dots,f_d$

$y_1,y_2,\dots,y_d$



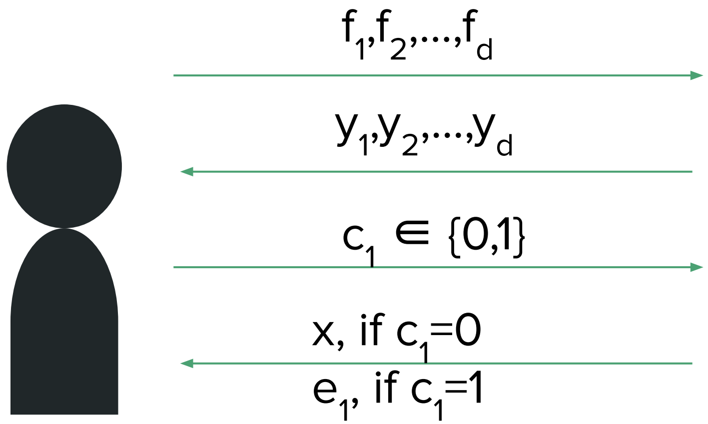
$\Sigma_x|x\rangle|f_1(x)\rangle$, $\Sigma_x|x\rangle|f_2(x)\rangle$,..., $\Sigma_x|x\rangle|f_d(x)\rangle$

$\Rightarrow (|x_0^1\rangle + |x_1^1\rangle)|y_1\rangle,\dots,(|x_0^d\rangle + |x_1^d\rangle)|y_d\rangle$

Remote Server

PoQ + “Pointer Chasing” = CVQD!

- Goal: Distinguish a depth-$(d + c^*)$ from a depth-d QC scheme
- Pointer chasing: a sequence of questions where the next question is determined by the last answer.



$\Sigma_x|x\rangle|f_1(x)\rangle, \Sigma_x|x\rangle|f_2(x)\rangle,\ldots, \Sigma_x|x\rangle|f_d(x)\rangle$

$\Rightarrow (|x_0^1\rangle + |x_1^1\rangle)|y_1\rangle,\ldots,(|x_0^d\rangle + |x_1^d\rangle)|y_d\rangle$

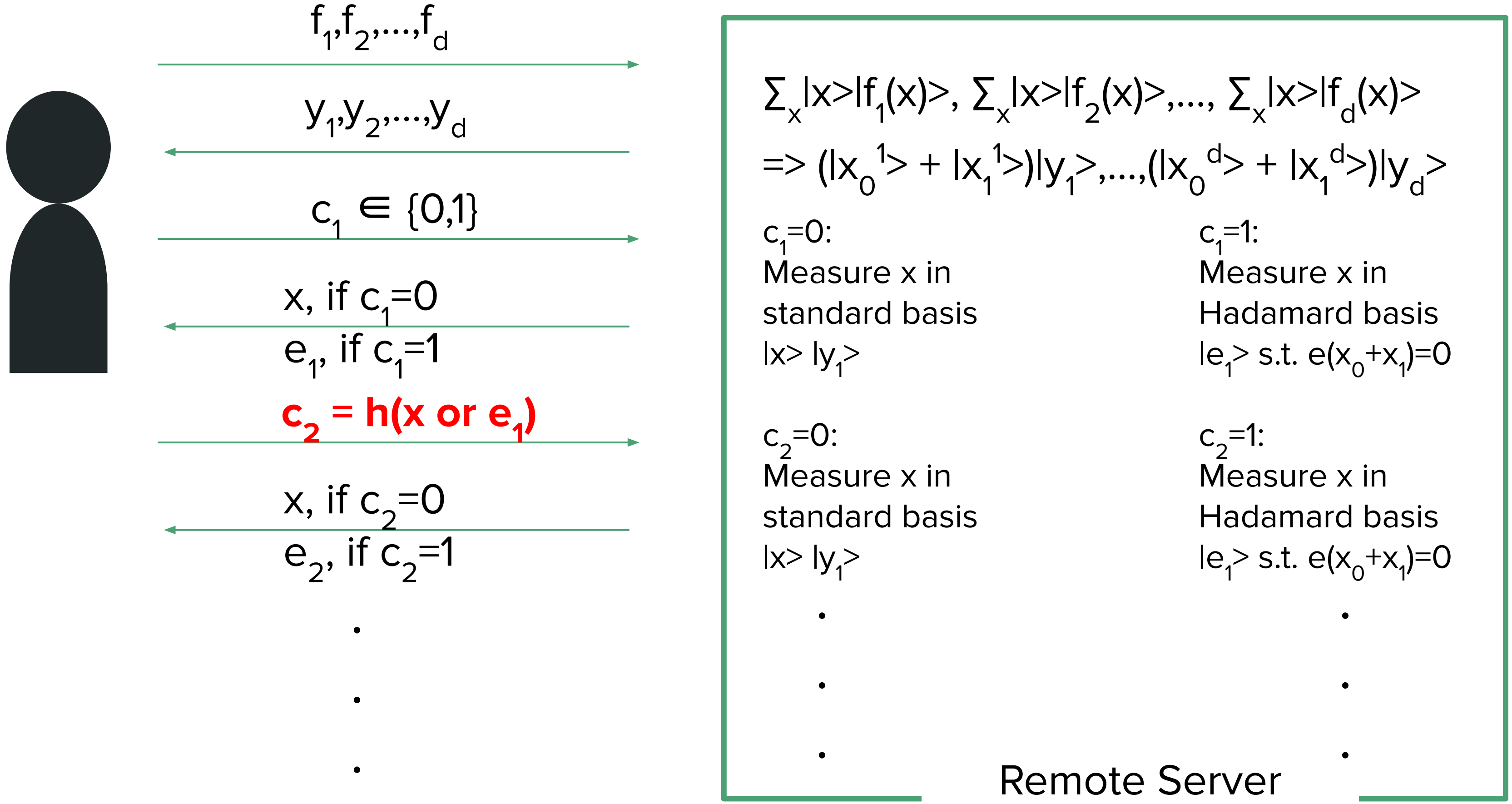
$c_1=0$:
Measure x in standard basis
$|x\rangle\ |y_1\rangle$

$c_1=1$:
Measure x in Hadamard basis
$|e_1\rangle$ s.t. $e(x_0+x_1)=0$

Remote Server

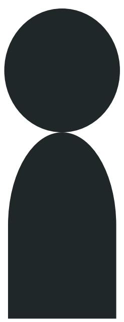
PoQ + "Pointer Chasing" = CVQD!

- Goal: Distinguish a depth-(d + c^*) from a depth-d QC scheme
- Pointer chasing: a sequence of questions where the next question is determined by the last answer.



PoQ + “Pointer Chasing” = CVQD!

- Goal: Distinguish a depth-(d + c*) from a depth-d QC scheme
- Pointer chasing: a sequence of questions where the next question is determined by the last answer.



$f_1, f_2, \ldots, f_d$ →

← $y_1, y_2, \ldots, y_d$

$c_1 \in \{0,1\}$ →

← x, if $c_1=0$ / e_1, if $c_1=1$

$\mathbf{c_2 = h(x \text{ or } e_1)}$ →

← x, if $c_2=0$ / e_2, if $c_2=1$

.

.

.

c_i is determined by the answer of the (i-1)-th round

Remote Server

$\Sigma_x |x\rangle|f_1(x)\rangle, \Sigma_x |x\rangle|f_2(x)\rangle, \ldots, \Sigma_x |x\rangle|f_d(x)\rangle$

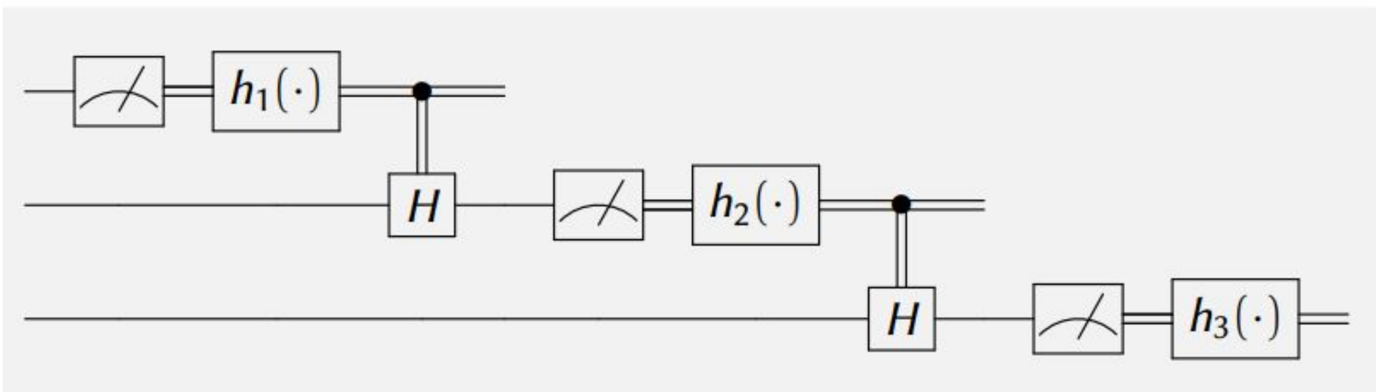
$\Rightarrow (|x_0^1\rangle + |x_1^1\rangle)|y_1\rangle, \ldots, (|x_0^d\rangle + |x_1^d\rangle)|y_d\rangle$

$c_1=0$:	$c_1=1$:
Measure x in standard basis	Measure x in Hadamard basis
$|x\rangle\ |y_1\rangle$	$|e_1\rangle$ s.t. $e(x_0+x_1)=0$
$c_2=0$:	$c_2=1$:
Measure x in standard basis	Measure x in Hadamard basis
$|x\rangle\ |y_1\rangle$	$|e_1\rangle$ s.t. $e(x_0+x_1)=0$
.	.
.	.
.	.

Completeness and Soundness

Completeness: ∃(d+c*)-QC that can win the game!

- Need c* depth to evaluate $\sum_x |x\rangle|f(x)\rangle$
- (i-1)-th answer determines i-th challenge
  - i-th qubit keeps coherence until all previous i-1 rounds complete



Soundness: ∀d'-QC with d'<d can't win the game breaks AHB

- Pigeon holes:
  - **1** quantum depth can only be used in **1** round
  - **d'** depth vs **d** rounds => ∃ rounds that are executed classically =>one can use these rounds to break AHB

First CVQD Protocol

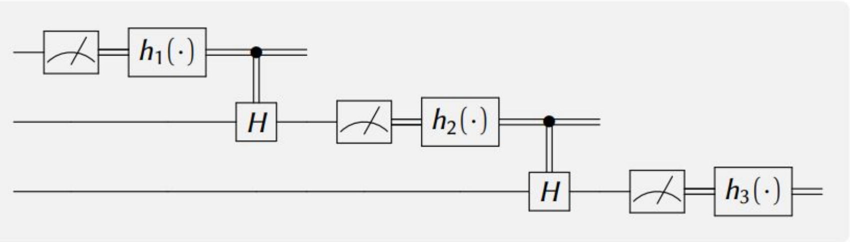
Theorem 1 [Chia-Hung 22]: ∃ CVQD protocols which distinguishes **d-depth quantum circuit** from **d+c*** for any d

Protocol A

1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
   a. Server's quantum depth > **d+c*** => Verifier accepts
   b. Server's quantum depth ≤ **d** => Verifier rejects
3. The remote server can **use classical computation** to cheat!

First CVQD Protocol

Theorem 1 [Chia-Hung 22]: ∃ CVQD protocols which distinguishes **d-depth quantum circuit** from **d+c*** for any d



Protocol A

1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
   a. Server's quantum depth > **d+c*** => Verifier accepts
   b. Server's quantum depth ≤ **d** => Verifier rejects
3. The remote server can **use classical computation** to cheat!

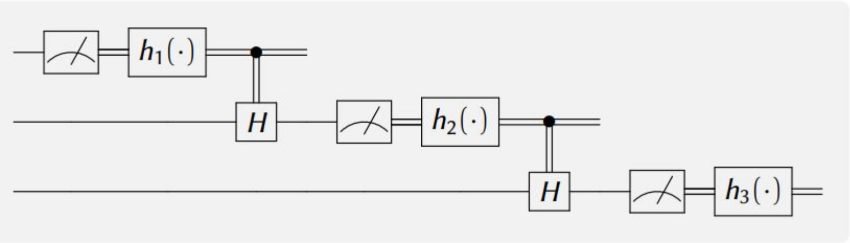
Disadvantages

- The separation **(d v.s. d+c*) is not optimal**
- Honest server **requires >c*-depth** to implement the protocol
- Honest server needs to be able to implement **d-QC scheme**

First CVQD Protocol

Theorem 1 [Chia-Hung 22]: ∃ CVQD protocols which distinguishes **d-depth quantum circuit** from **d+c*** for any d

Protocol A



1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
   a. Server's quantum depth > **d+c*** => Verifier accepts
   b. Server's quantum depth ≤ **d** => Verifier rejects
3. The remote server can **use classical computation** to cheat!

Disadvantages

- The separation **(d v.s. d+c*) is not optimal**
- Honest server **requires >c*-depth** to implement the protocol
- Honest server needs to be able to implement **d-QC scheme**

Question: Can we solve these issues?

Protocol B

Disadvantages of Protocol A

- The separation **(d v.s. d+c*) is not optimal**
- Honest server **requires >c*-depth** to implement the protocol
- Honest server needs to be able to implement **d-QC scheme**

Protocol B

Protocol B

- The separation **(d v.s. ~~d+c*~~) is not optimal** **d+3**
- ~~Honest server **requires >c*-depth** to implement the protocol~~
- ~~Honest server needs to be able to implement **d-QC scheme**~~ **d-depth circuit with classical post processing**

Protocol B

Protocol B

d+3

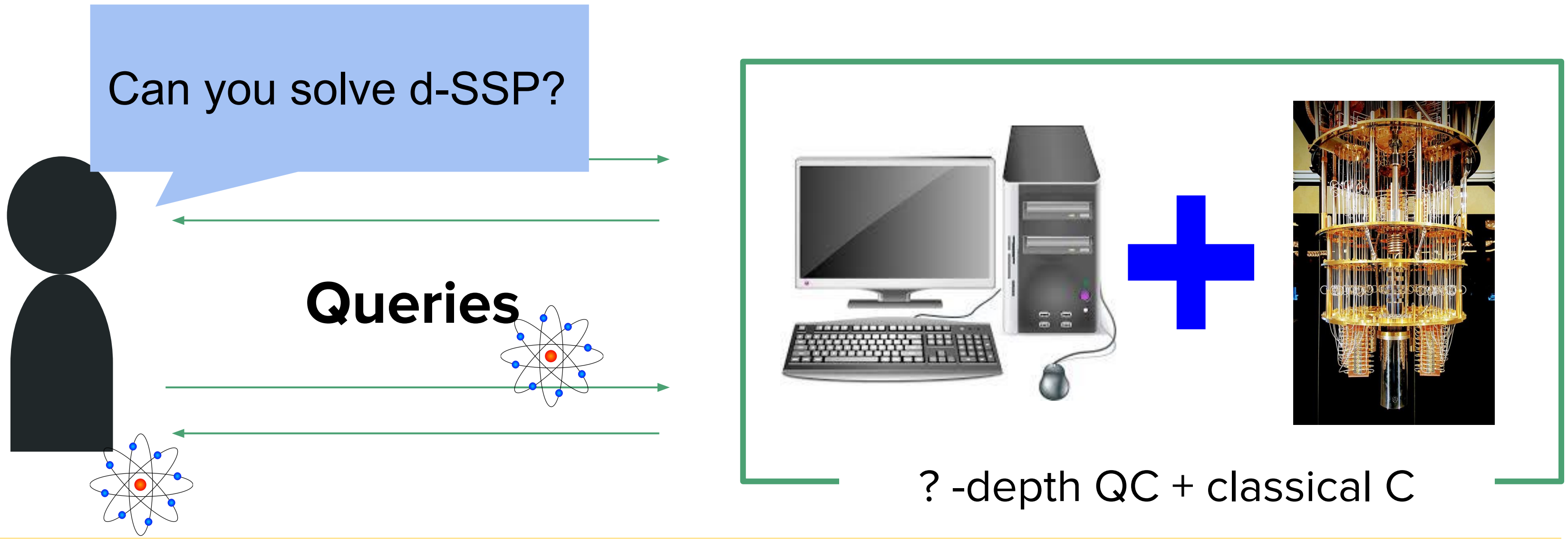
d-depth circuit with classical post processing

- The separation **(d v.s. ~~d+c*~~) is not optimal**
- ~~Honest server **requires >c*-depth** to implement the protocol~~
- ~~Honest server needs to be able to implement **d-QC scheme**~~

Idea:

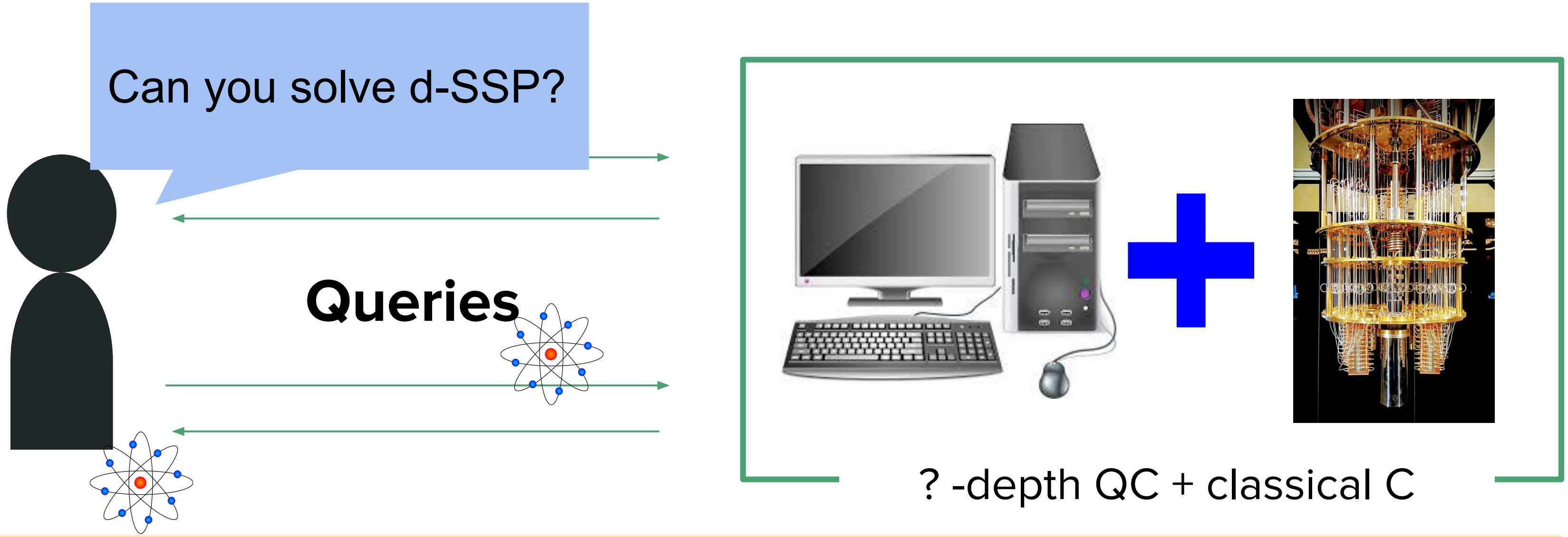
Revisit the first failed attempt (the protocol from d-SSP problem) and dequantize the protocol via **non-local game**

Need to “Dequantize” the Protocol



How to make the verifier and the protocol classical?

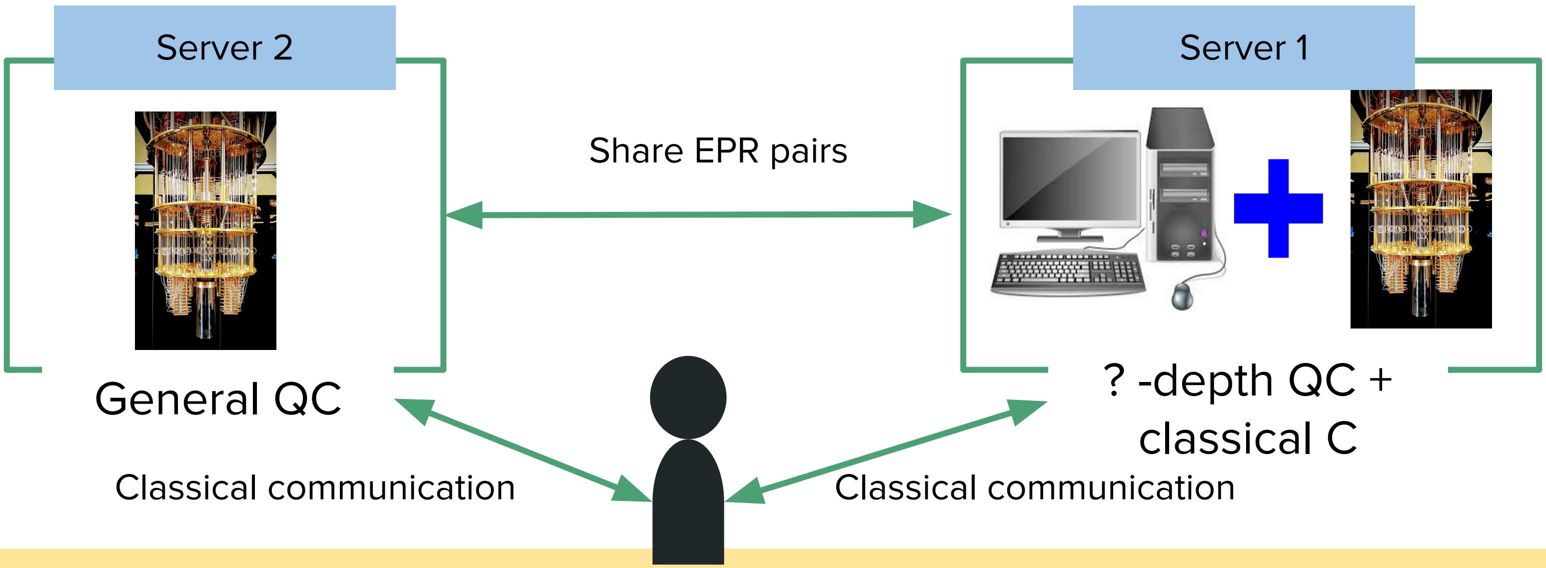
Need to "Dequantize" the Protocol



How to make the verifier and the protocol classical?

Idea: Delegate the "quantum computation" to another server!

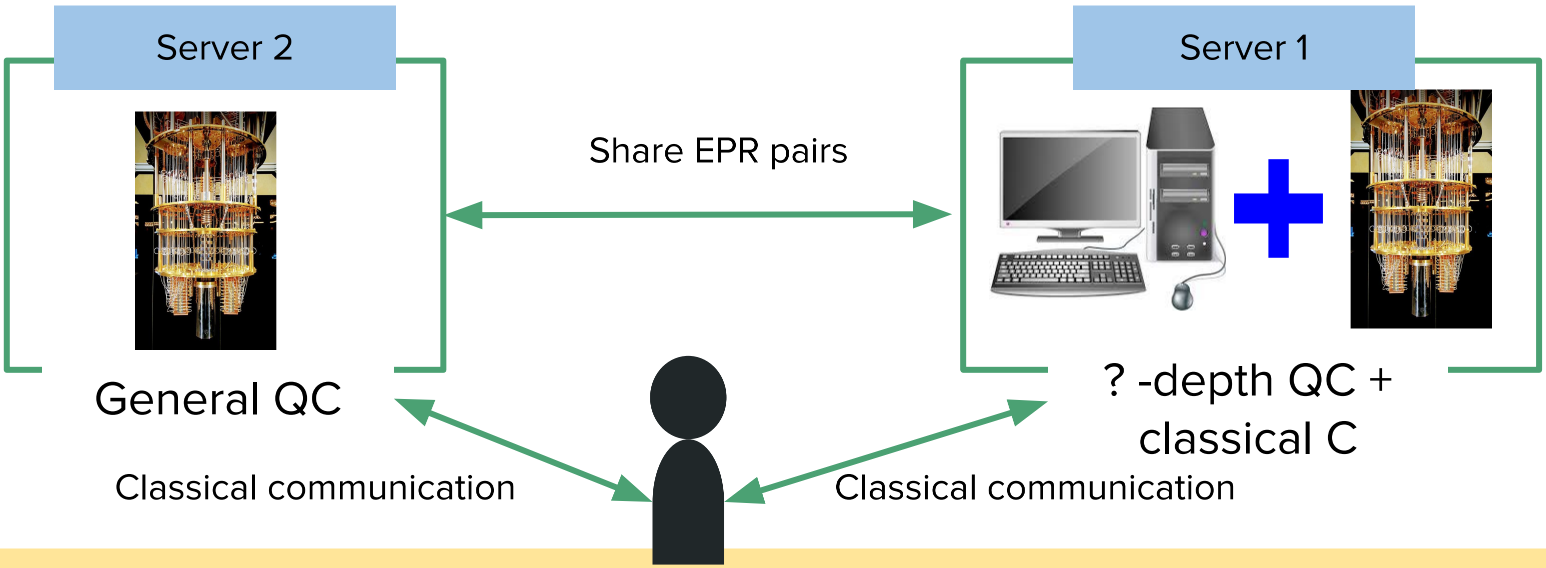
Need to “Dequantize” the Protocol



How to make the verifier and the protocol classical?

Idea: Delegate the “quantum computation” to another server!

Need to “Dequantize” the Protocol

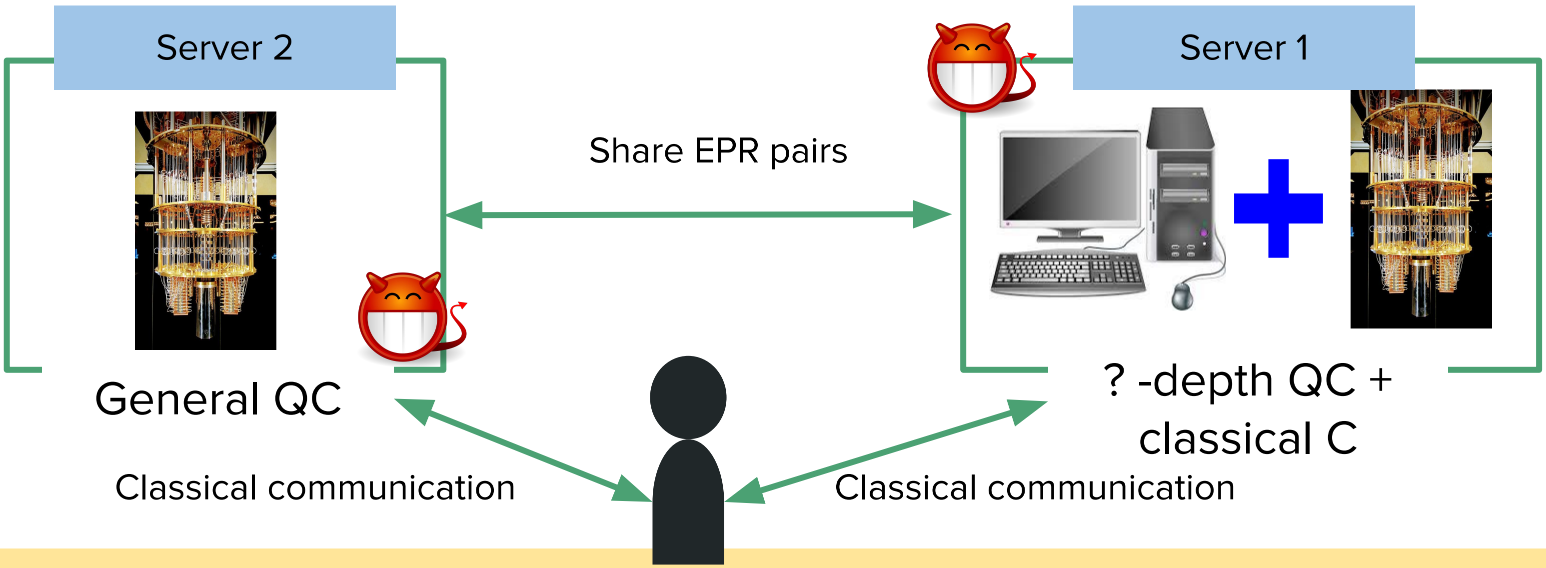


How to make the verifier and the protocol classical?

Idea: Delegate the “quantum computation” to another server!

- Server 2 implements the **quantum oracle**

Need to “Dequantize” the Protocol

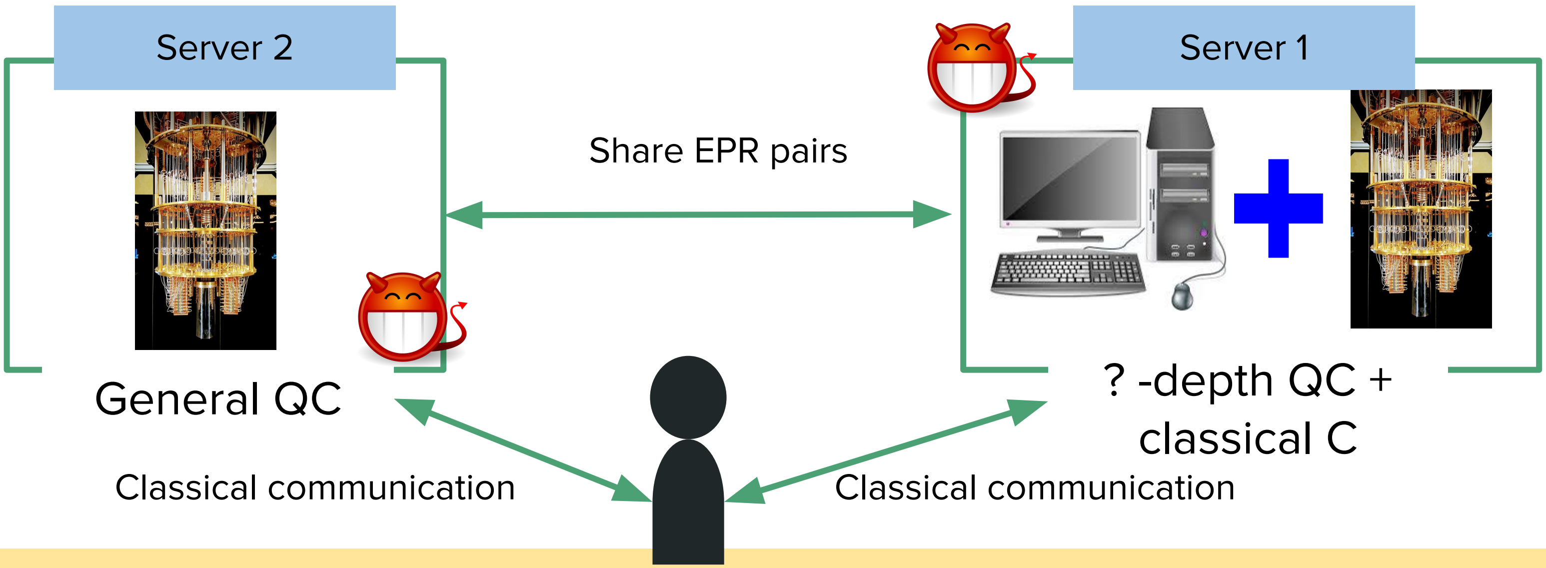


How to make the verifier and the protocol classical?

Idea: Delegate the “quantum computation” to another server!

- Server 2 implements the **quantum oracle**
- Server 1 and 2 **share EPR pair in the beginning** and **cannot talk**
    - The two servers are malicious and want to cheat

Need to “Dequantize” the Protocol

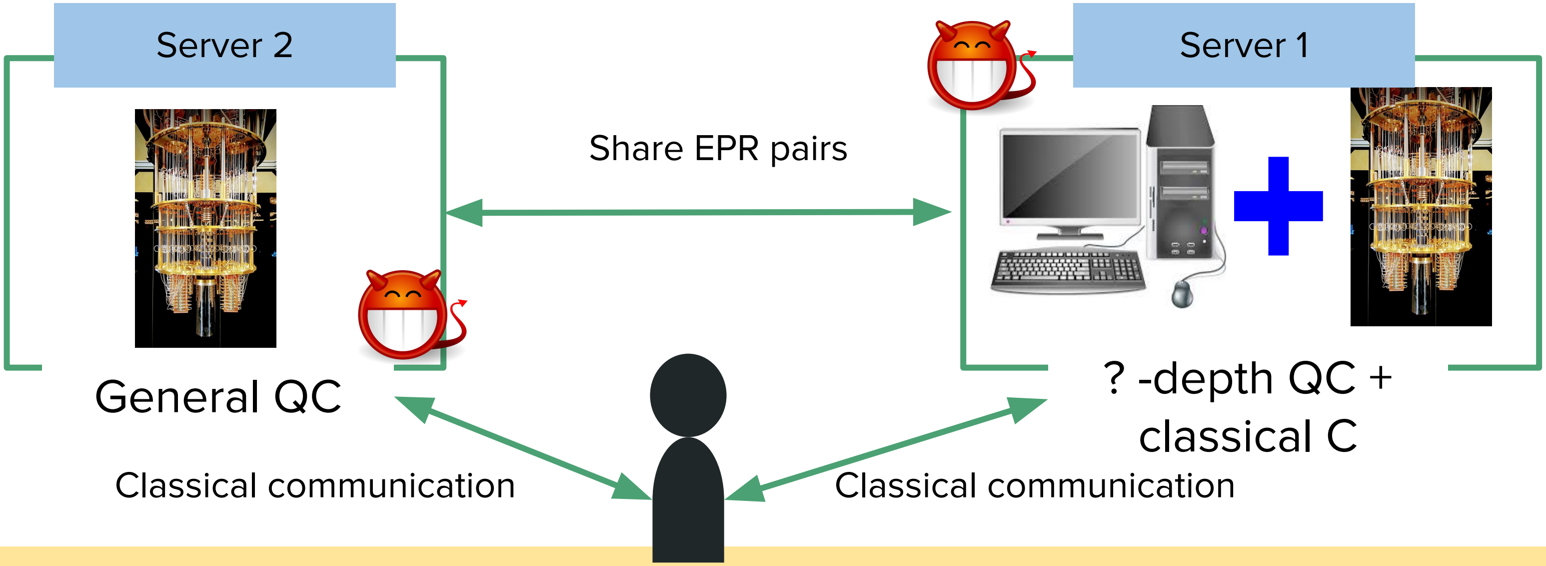


How to make the verifier and the protocol classical?

Idea: Delegate the “quantum computation” to another server!

- Server 2 implements the **quantum oracle**
- Server 1 and 2 **share EPR pair in the beginning** and **cannot talk**
  - The two servers are malicious and want to cheat
- Server 1 **teleports** queries to 2 by the EPR pairs, and vice versa
  - Pauli correction sent to the verifier via classical channel

Need to “Dequantize” the Protocol



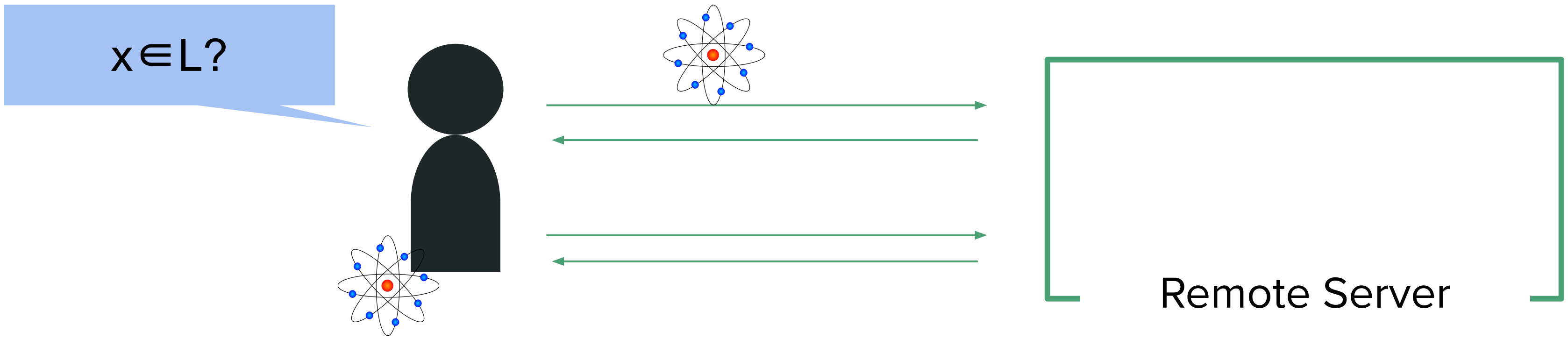
How to make the verifier and the protocol classical?

Idea: Delegate the “quantum computation” to another server!

- Server 2 implements the **quantum oracle**
- Server 1 and 2 **share EPR pair in the beginning** and **cannot talk**
  - The two servers are malicious and want to cheat
- Server 1 **teleports** queries to 2 by the EPR pairs, and vice versa
  - Pauli correction sent to the verifier via classical channel
- Verifier uses **additional test** to catch cheating servers

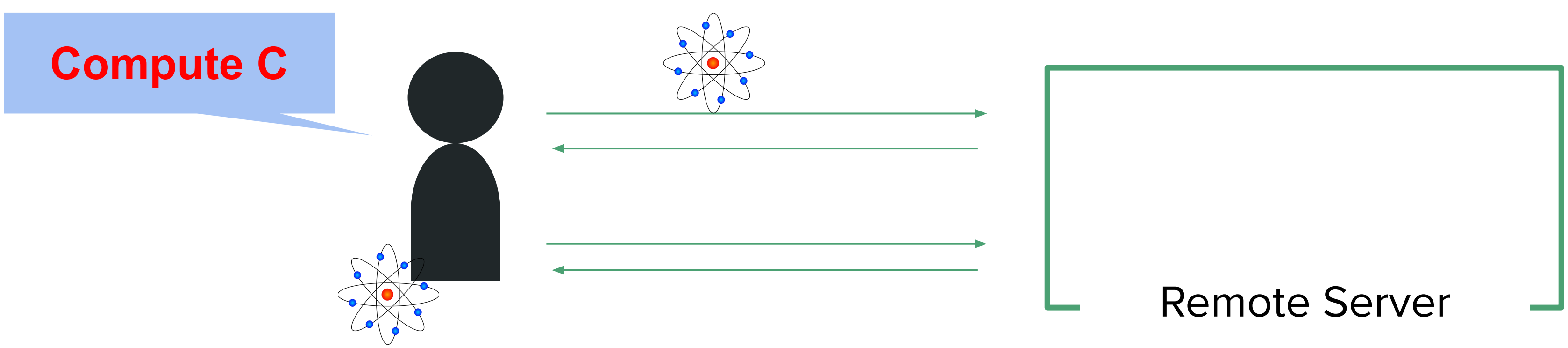
Sketch of the Proof

- The Broadbent protocol for **BQP** [Broadbent '18]
  - Three types: Comp, X-, and Z-test **(indistinguishable from server's view)**
  - Tests for detecting bit-flip (X-test) and phase-flip (Z-test).



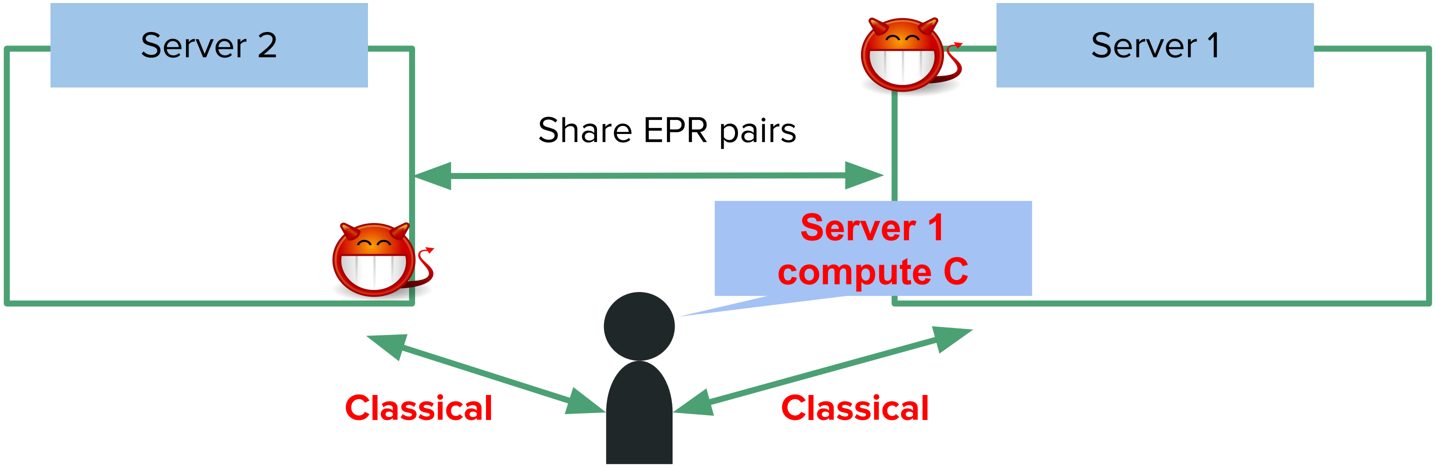
Sketch of the Proof

- The Broadbent protocol for **BQP** [Broadbent ’18]
  - Three types: Comp, X-, and Z-test **(indistinguishable from server’s view)**
  - Tests for detecting bit-flip (X-test) and phase-flip (Z-test).
- Theorem [Chia-Hung ’22]
  - For efficient circuit C, if server wins w.h.p., it implements a channel $C' \approx C$.
  - Use X-test and Z-test to check if the entire output state has error.

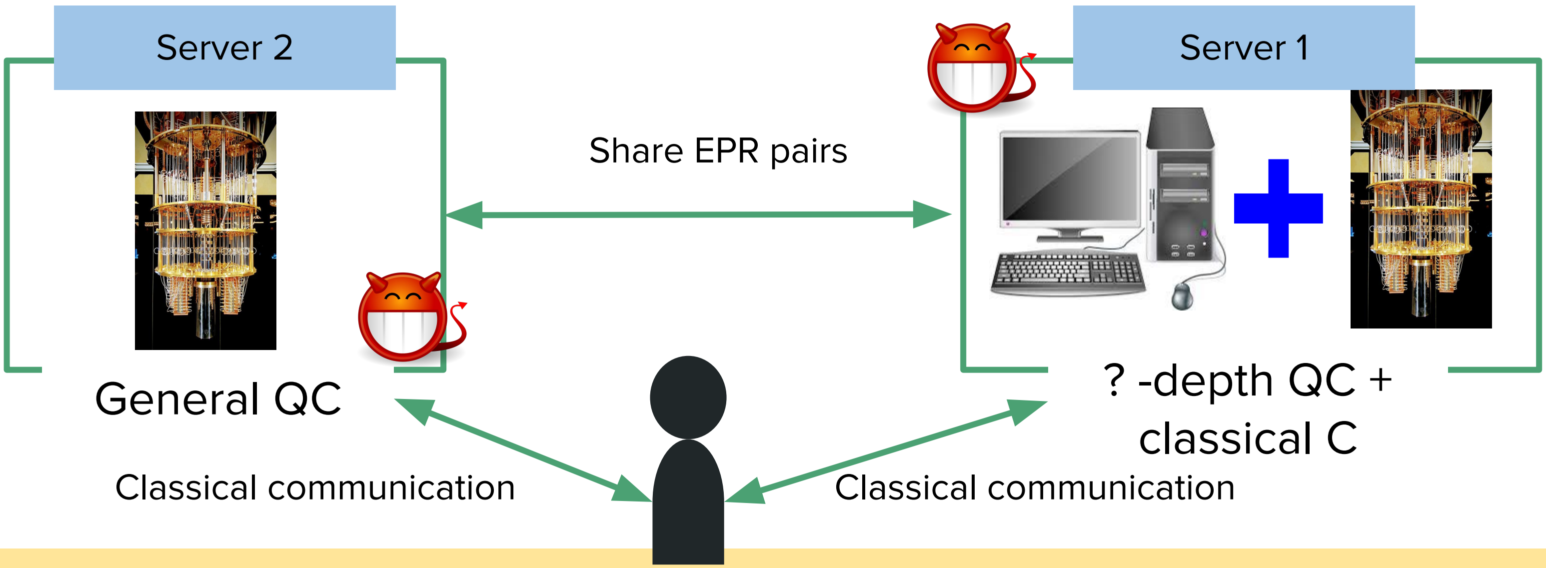


Sketch of the Proof

- The Broadbent protocol for **BQP** [Broadbent '18]
  - Three types: Comp, X-, and Z-test **(indistinguishable from server's view)**
  - Tests for detecting bit-flip (X-test) and phase-flip (Z-test).
- Theorem [Chia-Hung '22]
  - For efficient circuit C, if server wins w.h.p., it implements a channel C' ≍ C.
  - Use X-test and Z-test to check if the entire output state has error.
- Apply the verifier-on-a-leash protocol [Coladangelo, Grilo, Jeery, Vidick '19]:
  - ∃ two-player test which certifies Clifford measurements: if the player succeeds w.h.p., then the measurements is correct.



Need to “Dequantize” the Protocol



How to make the verifier and the protocol classical?

Idea: Delegate the “quantum computation” to another server!

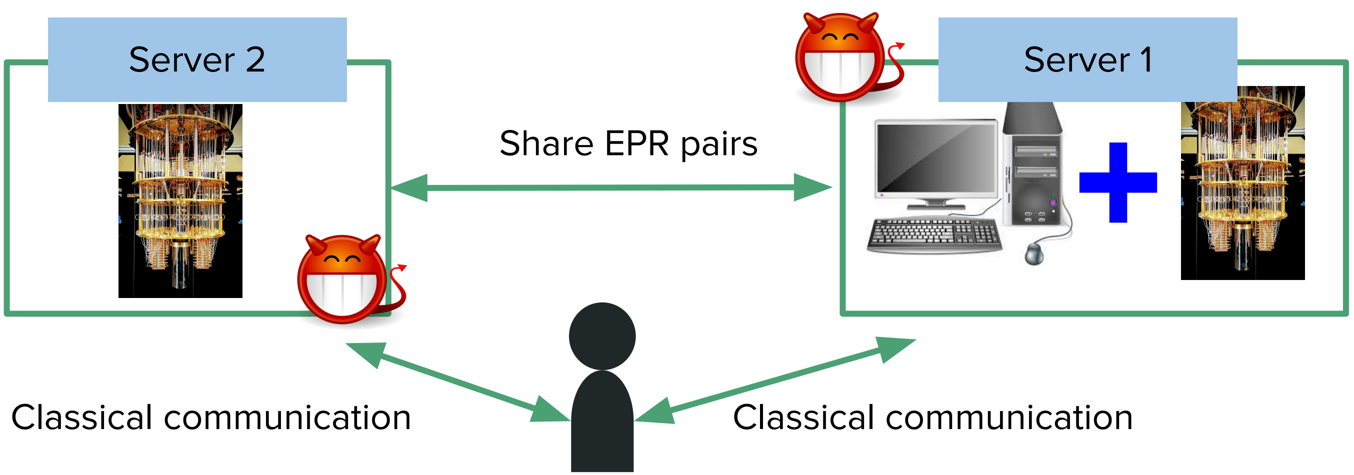
- Server 2 implements the **quantum oracle**
- Server 1 and 2 **share EPR pair in the beginning** and **cannot talk**
  - The two servers are malicious and want to cheat
- Server 1 **teleports** queries to 2 by the EPR pairs, and vice versa
  - Pauli correction sent to the verifier via classical channel
- Verifier uses **additional test** to catch cheating servers

Recap: Classical Verification of Quantum Depth

Theorem 2 [Chia-Hung 21]: There exists a two-prover classical protocol that satisfies the following:

1. The classical verifier accepts if the server has ≥ **d+3** quantum depth
2. The classical verifier rejects if the server has ≤ d quantum depth

- Unconditionally secure (inefficient) / Nearly optimal separation



Two-Prover CVQD

Theorem 1 [Chia-Hung 22]: ∃ two CVQD protocols where

- **Protocol A** distinguishes **d-depth quantum circuit** from **d+c*** for any d
- **Protocol B, a two-prover protocol,** distinguishes **d-depth quantum circuit** from **d+3** for any d

Protocol B

1. **Classical** verifier and **classical** protocol
2. Recognize server's quantum depth
    a. Server's quantum depth > **d+3** => Verifier accepts
    b. Server's quantum depth ≤ **d** => Verifier rejects
3. The server 1 only needs **d-depth circuit + classical post processing**

Unconditionally secure (inefficient) / Nearly optimal separation

Disadvantage of Protocol B
Protocol B **requires another dishonest quantum prover** to help...

Open Questions

Open question 1: Classical verification of quantum **resources**, e.g., quantum memory, quantum volume, etc.

Open question 2: Improved CVQD protocol that can be implemented **on near-term devices**

Open question 3: Separations between hybrid quantum-classical computation (d-CQ and d-QC schemes) **in the plain model**

Open Questions

Open question 1: Classical verification of quantum **resources**, e.g., quantum memory, quantum volume, etc.

Open question 2: Improved CVQD protocol that can be implemented **on near-term devices**

Open question 3: Separations between hybrid quantum-classical computation (d-CQ and d-QC schemes) **in the plain model**

Thank you!