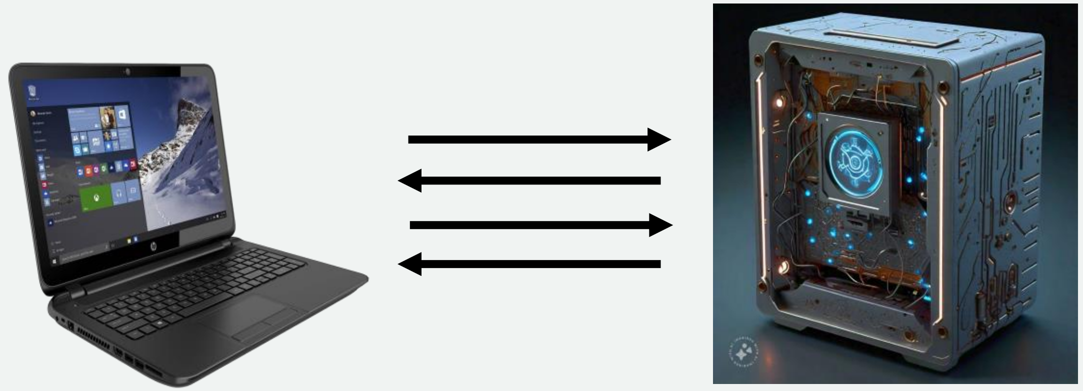
Quantum Logspace Computations are Verifiable

Anish Banerjee

Jaspreet Randhawa

Based on [GRZ23]



Classical Verifier **Quantum Prover**

History

Quantum computers can execute computations beyond the range of classical computers (hopefully!)

Can’t “predict and verify”!

Prior Work

[Got04] : Is it possible for an **efficient classical verifier** to verify the output of an **efficient quantum prover**?

[Mah23] : Yes!

- But her protocol is secure against **computationally bounded adversaries**, under **cryptographic assumptions**.

THIS WORK

Unbounded adversaries

A classical **logspace** verifier can verify the output of a quantum **logspace** prover

Non-interactive

UNITARY MATRIX POWERING

Input:

- Unitary matrix $\mathbf{M}_{\mathbf{nxn}}$
- Parameter K
- Projector $\mathbf{\Pi}$

Promise:

- $||\mathbf{\Pi}\mathbf{M}^{K}\mathbf{e}_{\mathbf{1}}||^{2} \geq 4/5$ or
- $||\mathbf{\Pi}\mathbf{M}^{K}\mathbf{e}_{\mathbf{1}}||^{2} \leq 1/5$

Output: Determine the case

Theorem:
Unitary Matrix Powering is logspace complete for **BQL**

Main Idea

Streaming Proof

– *a **space-bounded** algorithm with access to a **massive** stream of data*

– *verify a computation that requires large space, by communicating with a **powerful untrusted prover***

– *log(n) space verifier and poly(n) size proof*

Classical Verifier

Quantum Prover

Read Once

Write Once

Proof Tape

Main Claims

δ-Good Sequence

Consider the sequence: $\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_K$, (K=poly(n)) where $\mathbf{v}_i = \mathbf{M}^i \mathbf{e}_1$

A sequence $\mathbf{v'}_0, \mathbf{v'}_1, \cdots, \mathbf{v'}_K$ is δ-good if for all i

$$|| \mathbf{v}_i - \mathbf{v'}_i || \leq \delta$$

CLAIM1:

There exists a BQL prover which outputs a δ-good sequence of vectors.

[GRZ21] : A quantum logspace algorithm for powering matrices

CLAIM2:

There is a randomized logspace verifier which given "read-once" access to the stream of vectors accepts iff the stream is δ-good.

Conclusion

Quantum logspace computations can be verifiably checked by a classical logspace algorithm with **unconditional security**.

For any problem in BQL:

- Reduce it to UMP
- Use above protocol for verification

Thank you