

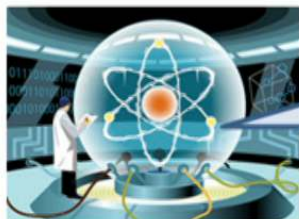
Title: Certified Randomness from Quantum Supremacy

Date: Jul 25, 2018 04:00 PM

URL: <http://pirsa.org/18070057>

Abstract: We describe a novel application for near-term quantum computers with 50-70 qubits: namely, generating cryptographic random bits, whose randomness can be certified even if the quantum computer is untrusted (e.g., has been backdoored by an adversary). Unlike schemes based on Bell inequality violation, ours requires only a single device able to solve classically hard sampling problems. Our protocol harvests the outputs of the sampling process and feeds them into a randomness extractor, while occasionally verifying the outputs using exponential classical time. We also compare to the beautiful independent work of Brakerski et al., who proposed a scheme for the same problem that has much more efficient verification, but that probably cannot be implemented on near-term devices. Paper still in preparation.

Certified Randomness from Quantum Supremacy



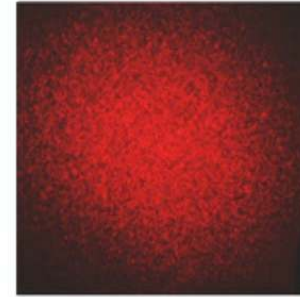
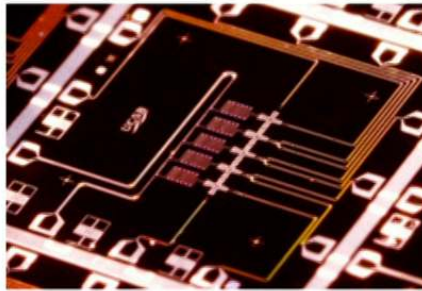
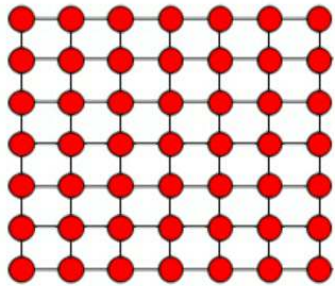
1101000011010011
1101101100110011
0001010010011010
0011111011110100



Scott Aaronson (UT Austin)

Perimeter Institute, Waterloo, July 25, 2018

Sampling-Based Quantum Supremacy



Theoretical foundations: Bremner-Jozsa-Shepherd (IQP), A.-Arkhipov 2011 (BosonSampling), A.-Chen 2017 (RCS)...

Showed that sampling the output distributions of various quantum systems is classically intractable under plausible assumptions...

My line for years: Exciting because you can do this with a NISQ device, and because it will refute Gil Kalai.

Clearly, obviously, it's useless in and of itself...

11010000110100111101101100110011000101001001

Certified Random Bits: Who Needs 'Em?

For private use:

Cryptographic keys (a big one!)

For public use:

Election auditing, lotteries, parameters for cryptosystems, zero-knowledge protocols, proof-of-stake cryptocurrencies...



Certified Random Bits: Who Needs 'Em?

For private use:

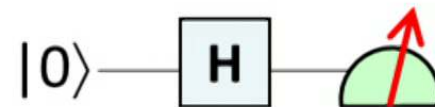
Cryptographic keys (a big one!)

For public use:

Election auditing, lotteries, parameters for cryptosystems, zero-knowledge protocols, proof-of-stake cryptocurrencies...



Trivial Quantum Randomness Solution!



Problem: What if your quantum hardware was backdoored by the NSA? (Like the DUAL_EC_DRBG pseudorandom generator was?) Want to trust a deterministic classical computer only

Earlier Approach: Bell-Certified Randomness Generation

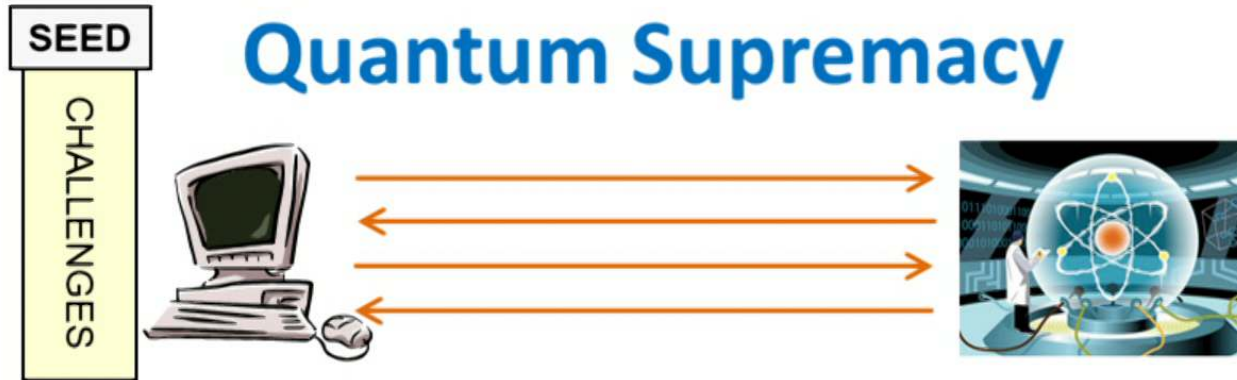
Colbeck and Renner, Pironio et al., Vazirani and Vidick, Coudron and Yuen, Miller and Shi...



Upside: Doesn't need a QC; uses only "current technology" (though loophole-free Bell violations are only ~2 years old)

Downside: If you're getting the random bits over the Internet, how do you know Alice and Bob were separated?

New Approach: Randomness from Quantum Supremacy



Key Insight: A QC can solve certain sampling problems quickly—but under plausible hardness assumptions, it can **only** do so by sampling (and hence, generating real entropy)

Upsides: Requires just a single device—perfect for certified randomness over the Internet. Ideally suited to NISQ devices

Caveats: Requires hardness assumptions and initial seed randomness. Verification (with my scheme) takes $\exp(n)$ time



INHERENTLY REQUIRES QM

Why? Because a classical server could always replace its randomness source by a pseudorandom one without the client being able to detect it

Indeed, our protocol requires certain tasks (e.g., finding heavy outputs of a quantum circuit) to be **easy** for QCs, and other tasks (e.g., finding the same heavy outputs every time) to be **hard** for QCs!

Applications



For the QC owner:
Private randomness



For those connecting over the cloud: Public randomness

The protocol does require pseudorandom challenges, but:

Even if the pseudorandom generator is broken later, the truly random bits will remain safe (“forward secrecy”)

Even if the seed was public, the random bits can be private

The random bits demonstrably weren’t known to *anyone*, even the QC, before it received a challenge (freshness)

The Protocol

1. The classical client generates n -qubit quantum circuits C_1, \dots, C_T pseudorandomly (mimicking a random ensemble)
2. For each t , the client sends C_t to the server, then demands a response S_t within a very short time

In the “honest” case, the response is a list of k samples from the output distribution of $C_t |0\rangle^{\otimes n}$

3. The client picks $O(1)$ random iterations t , and for each one, checks whether S_t solves “HOG” (Heavy Output Generation)
4. If these checks pass, then the client feeds $S = \langle S_1, \dots, S_T \rangle$ into a classical **randomness extractor**, such as GUV (Guruswami-Umans-Vadhan), to get nearly pure random bits

The HOG Problem

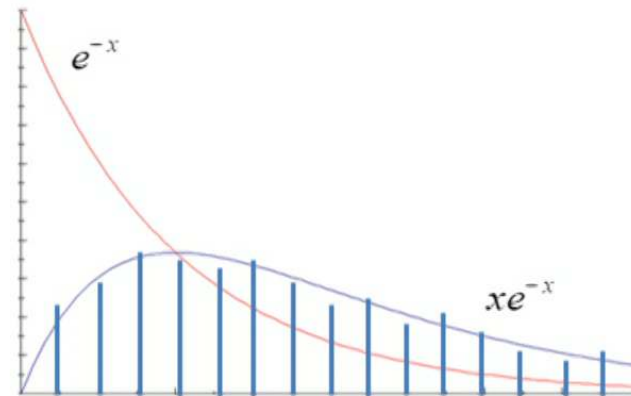
Given as input an n-qubit quantum circuit C, and parameters $b \in (1, 2)$ and $k \in \mathbb{N}$, output n-bit strings s_1, \dots, s_k such that

$$\sum_{i=1}^k \Pr[C \text{ outputs } s_i] \geq \frac{bk}{2^n}$$

We can verify that $\langle s_1, \dots, s_k \rangle$ solves HOG in $\sim 2^n$ classical time

For $b \in (1, 2)$ and large enough k, an ideal QC solves $\text{HOG}_{b,k}$ with probability close to 1, because of **Porter-Thomas speckle behavior** and the law of large numbers

$$\Pr[\text{success}] = e^{-bk} \sum_{j=0}^{2k-1} \frac{(bk)^j}{j!}$$



Long List Hardness Assumption (LLHA)

Arthur is given random access to a list of random n -qubit quantum circuits C_1, \dots, C_M , as well as n -bit strings s_1, \dots, s_M , where (say) $M=2^{3n}$. He's promised that either

- (1) the s_i 's were drawn uniformly at random, or
- (2) each s_i was drawn from the output distribution of C_i .

Then there's no **AM** protocol wherein Arthur sends Merlin an $n^{O(1)}$ -bit classical challenge, then Merlin sends back an $n^{O(1)}$ -bit reply that convinces Arthur it's case (2). Even if Arthur can do the verification using a $2^{0.49n}$ -time quantum computation, with $2^{0.49n}$ qubits of quantum advice.

Basic Theorem: Suppose LLHA holds. Let Q be a polynomial-time quantum algorithm that solves $\text{HOG}_{b,k}(C)$ with success probability at least q , averaged over all C . Then Q 's output distribution has at least

$$\frac{bq - 1}{b - 1} - o(1)$$

of its mass (again, averaged over all circuits C) on outputs with probability at most $2^{-0.49n}$

Proof Idea: Using a “low-entropy” quantum algorithm Q to solve HOG, we design an **AM** protocol, based on Stockmeyer approximate counting, that violates LLHA—one where Merlin points Arthur to various high-probability outputs of Q to convince him he's in case (2)

Pseudorandom Function Assumption (PRFA)

There's a family of efficiently-computable Boolean functions, $g_k: \{0,1\}^n \rightarrow \{0,1\}$, parameterized by an $O(n)$ -bit seed k , such that for every 3^n -time quantum algorithm Q ,

$$\left| \Pr_k \left[Q^{g_k} \text{ accepts} \right] - \Pr_g \left[Q^g \text{ accepts} \right] \right| \leq 0.01.$$

Moreover, this is true even if Q is a **PDQP** algorithm (A. et al. 2014): a quantum algorithm that can contain both ordinary measurements and “non-collapsing” measurements

Main Result

Suppose LLHA and PRFA both hold, and that the server does at most $n^{O(1)}$ quantum computation per iteration. Suppose also that we run the protocol, for $T \leq 2^n$ steps, and the client accepts with probability $> \frac{1}{2}$. Then conditioned on the client accepting, the output bits S are $1/\exp(n^{\Omega(1)})$ -close in variation distance to a distribution with **min-entropy** $\Omega(Tn)$.

$$H_{\min}(D) := \min_x \log_2 \frac{1}{\Pr_D[x]}$$

Which means: the extractor will output $\Omega(Tn)$ bits that are exponentially close to uniform

Hard part: show **accumulation** of min-entropy across the T iterations. E.g., rule out that the samples are correlated

FAQ for Experimentalists

How many qubits? Maybe 40-70—enough to see a clear quantum/classical separation, but not so many that classical verification becomes impossible

What circuit depth? Large enough that tensor network and other clever methods fail, and LLHA is plausible

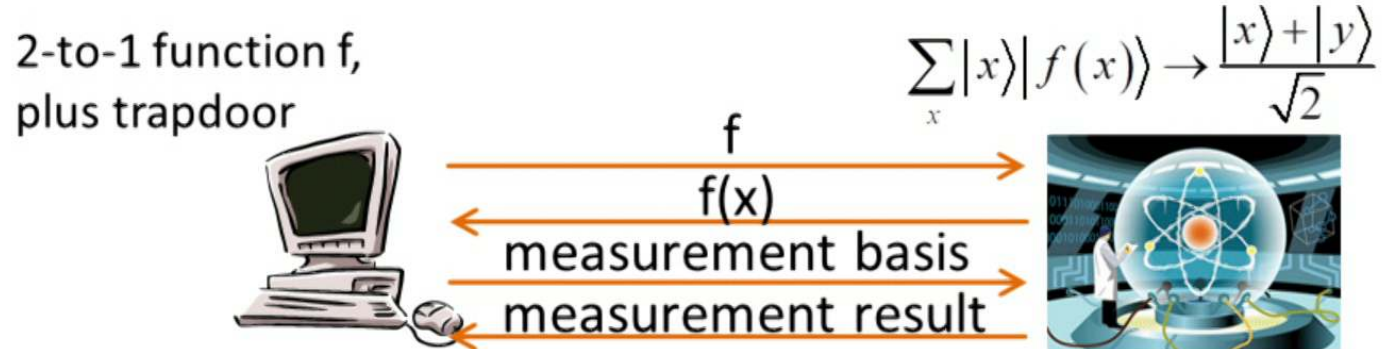
What fidelity? Enough that $\text{HOG}_{b,k}$ is getting solved for some b,k , with a high enough probability that our theorem guarantees min-entropy generation

What's the bit rate? The more error, the larger the k we'll need, hence the lower the bit rate. The need to keep swapping out the circuit C_t could also slow things down. But a few hundred bits per second seems plausible...

Independent Approach

Brakerski, Christiano, Mahadev, Vazirani, Vidick arXiv:1804.00640

Method for a QC to generate random bits, assuming the quantum hardness of breaking lattice-based cryptosystems



Huge advantage of the BCMVV scheme: Polynomial-time classical verification!

Huge advantage of mine: Can be run on NISQ devices!

Future Directions

Can we get polynomial-time classical verification and NISQ implementability at the same time?

Can we get more and more certified randomness by sampling with the **same** circuit C over and over? Would greatly improve the bit rate, remove the need for a PRF

Can we prove our scheme sound under less boutique complexity assumptions (without the quantum advice, non-collapsing measurements, long list of circuits...)?

Can we prove our scheme sound even against adversaries that are entangled with the QC?

Conclusions

Certified randomness generation: the **most** plausible application of a near-term quantum computer?

Not only can we do it with ≤ 70 qubits, we don't **want** more

Requires a sampling problem: problems with definite answers (like factoring) are useless

No expensive encoding needed; can fully exploit hardware (just like with quantum supremacy demos)

With randomness generation, all the weaknesses of sampling-based quantum supremacy have become strengths!