Reproducing Google's Quantum Supremacy Experiments on IBM Quantum Hardware

Matteo Bonacini

Abstract—Quantum computers are fundamentally faster than classical ones. In 2019, Google made an effort to prove just that by exploiting some properties of random quantum circuits. In this paper, I explain the details and the controversies behind their study and I show how the same experiment could be adapted to different modern hardware.

1 Introduction

In 2019, researchers from Google AI and Quantum published a paper titled *Quantum supremacy using a programmable superconducting processor* [1]. The publication was received with some criticism from the scientific community due to apparent methodological flaws and due to its potentially misleading title. The harshest response came from IBM [2], who claimed that Google's computational complexity estimates were off by orders of magnitude. In 2021, an independent group of researchers published a new paper claiming *quantum supremacy* [3], which applied IBM's proposed methodology to Google's 2019 paper and obtained more plausible results.

Regardless of the criticism received, Google's work is still relevant as it put a point in favor towards the viability of quantum computers. In this report, I will analyze the work made by Google as well as the main criticisms they received, and then draw conclusion for the current state of quantum supremacy. I will also show what it would take to reproduce Google's experiment on different quantum hardware. To best explain myself, I will start off my analysis with a brief discussion on the meaning of quantum supremacy and on random quantum circuits.

1.1 Quantum supremacy

The term *quantum supremacy* was coined by Preskill[1]in his 2012 paper *Quantum Computing and The Entanglement Frontier* [4], in which he highlighted several structural differences between the classical and quantum computation paradigms. The main takeaways that interest us are:

1) There exist a class of algorithms whose completion time scales polinomially as a function of the number of inputs. These can be easily solved and implemented by classical computes.
2) Algorithms which scale exponentially *cannot* be easily solved by a classical computer (NP-hard problems).
3) There exist algorithms that can solve NP-hard problems in polynomial time, but they require quantum hardware to be run.

Thus, quantum supremacy refers to the fact that quantum computers would be capable of solving a large class of problems with an exponential speedup.

Now, one must be careful when talking about complexity classes, as there may be some ambiguity (proving that P≠NP is the problem of the millennium); a broader picture is illustrated in Figure 1 [5]. Caveats aside, from my understanding of his writing, Preskill (along with most experts in the field) would say that we have reached quantum supremacy only after seeing a *NP-complete algorithm which we can classically verify in polynomial time* being run in polynomial time on a quantum computer. An example would be the factorization of large numbers which, in the general case, cannot be done easily (luckily, I would say, as this would break most modern cryptography!).

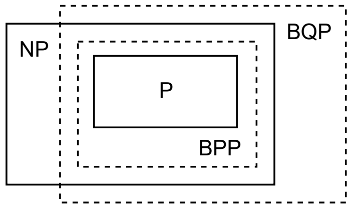


Figure 1: General overview of complexity classes' relationships. Each class represents a different kind of problem: P = solvable classically; NP = not solvable classically; BPP = solvable classically using a random algorithm with defined success rate; BQP = solvable using a quantum computer. We are not sure where exactly the BQP and BPP classes lie and the dashed rectangles in the picture are our current best guesses. There is also no strong proof that NP is greater than P.

1.2 Random quantum circuits

A quantum circuit can be defined by a unitary operator U that transforms an arbitrary initial state $|\psi_0\rangle$ into a final state $|\psi\rangle$:

$$|\psi\rangle = U\,|\psi_0\rangle\,.$$

In practice, when writing a quantum circuit, it is most common to write it as a sequence of quantum *gates*, unary operators that act only on a subspace of the Hilbert space in which $|\psi\rangle$ lives. It is proven that there exist finite sets of quantum gates that can be combined to produce any unitary operator; these are called *universal sets* [5]. A basic definition of a random quantum circuit is the following:

Definition 1. A circuit is said to be *random* when U is uniformly chosen at random among all the possible unitary matrices.

In practice, we can generate a random circuit by layering a large number of gates chosen at random from a universal set. The convergence to randomness is exponentially slow

1. John Phillip Preskill (1953) is an American theoretical physicist and professor at the California Institute of Technology, where he is also the director of the Institute for Quantum Information and Matter.

in the number of gates, but there are techniques that allow us to efficiently obtain good approximations [6, 7].

There are two questions that concern us.

1) Mathematically, how can we sample unitary matrices *uniformly*?
2) What is the distribution for the final measurement probabilities $P(x) = |\langle x|\psi\rangle|^2$ of a random circuit?

The answers come from *random matrix theory*, and we will now see a brief summary of the most important concepts.

1) The Haar measure

We want to samples matrices $U \in \mathbb{U}^{2n\times 2n}$ uniformly, so that $P(U) = P(U + \Delta U)$. A thorough explanation of the subject would require deep knowledge of measure theory, which I do not possess. I am going to limit the discussion to the conceptual results.

The idea is that we need to define an integration measure on the space $\mathbb{U}^{2n\times 2n}$ that weighs each point equally. Doing so is non-trivial for arbitrary n, but I can provide an example for the case $n = 1$ (below). This measure takes the name of *Haar* measure, from the person who first introduced it in 1933 [8] in the framework of group theory. A (bare-bones) definition of Haar measure follows.

Definition 2. A Haar measure $\mu(g)$, where $g \in G$ is an element of the space over which we are measuring, is a measure with two additional conditions:

1) $\int \mu(g) = 1$ (Normalization of probability)
2) $\mu(A) = \mu(gA))$, $A \subseteq G$, $gA = \{gx/x \in A\}$ (Left invariance).

In particular the *left invariance* property ensures that the measure is uniform in G with respect to a translation by its elements.

Now, in order to provide an example for $1x1$ matrices, consider $U = (z)$ with $z \in \mathbb{C}$. The unitariety condition reads: $U^*U = 1$, from which $|z|^2 = 1$. This defines the unit circle in the complex plane; thus, we can write our matrix as $U = (e^{i\phi})$, $\phi \in [0, 2\pi[$. A properly defined Haar measure in this space is given by $\mu(\phi) = d\phi/V$, with V being the diameter of the circumference: $V = \int_0^{2\pi} d\phi = 2\pi$.

2) The Porter-Thomas distribution

When we measure the output of a n-qbit random circuit, we get a number x as a result, with $x \in [0, 2^n[\cap\mathbb{N}$. In the literature, it is more common to work with the binary representation of x, which is referred to as *bitstring*. If the quantum circuit we are measuring effectively samples from the Haar distribution, it can be proven that the *distribution of measurement probabilities* will follow the Porter-Thomas distribution:

$$P(p) = Ne^{-Np}, \tag{1}$$

where $N = 2^n$ is the dimension of the Hillbert space. This result is valid only if the computation is performed in a noiseless environment. If we introduce gate errors in the circuit, the bitstring distribution tends to a uniform one. The quantum supremacy experiment was engineered based on these results (which are proven in [9]).

One important consideration on the notation of (1): the probability distribution is not over the *bitstrings*, but is over the *probabilities of measuring any particular bitstring*. To get this distribution experimentally, we would first need to sample the same circuit $\mathcal{O}(2^n)$ times. From this, we get the probabilities $p(x)$ of observing a particular bitstring. Grouping these into bins would yield an apporximate version of (1).

2 QUANTIFYING SUPREMACY

In this section, I will explain the most critical points of Google's paper [10] and of [9]. Before moving on, please note that the argument for supremacy they provide is essentially based on a circular argument. We will see how Google managed to get around this.

2.1 Theoretical background

In [9] they propose the following benchmark for quantum supremacy. Suppose I have a quantum n-qbit system prepared in the state $|\phi_0\rangle$, and I apply a random circuit to it: $|\psi\rangle = U|\psi_0\rangle$. Suppose that I can, somehow, know the full state vector $|\psi\rangle$. Then, suppose I have access to both a quantum and a classical computer that sample a set $S = \{|x_1\rangle, \ldots, |x_m\rangle\}$ of m bitstrings from an *imperfect* (be it noisy or approximated) realization of U. Now, define the *linear cross-entropy benchmarking (XEB) fidelity* as

$$F = 2^n \frac{1}{m} \sum_{|x_i\rangle \in S} |\langle x_i|\psi\rangle|^2 - 1.$$

Their derivation shows that when the x_i are sampled according to a perfect realization of U, $F = 1$, and when the sampling is a uniform random guess, $F = 0$. Also, F represents the probability that the computation has happened *without any error*. They claim that quantum supremacy is achieved when

$$1 \geq F_q > F_{cl} > 0,$$

where F_q is obtained by sampling with a quantum computer and F_{cl} is obtained by sampling using the best possible classical software and hardware. F_q here is also used to check that the quantum computer is still performing somewhat real computations, by imposing $F_q > 0$. Another property of F that makes it a great tool is that its uncertainty scales polinomially in the number of samples: $< F^2 >\propto 1/N$. This is good, as sampling a quantum circuit an exponential number of times is *hard* even for quantum computers.

Now, let us address the elephant in the room. Cross entropy may be a great tool for benchmarking purposes (e.g. for estimating single-gate error rates), but in order to compute it we need to know the full system state. This can be done only through a classical simulation (or an exponential number of quantum circuit runs, which would take roughly the same resources). This means that *just being able to compute F for a large circuit would disprove the supremacy claim*. Google did come up with some arguments to make up for this fallacy; we will see them in the following sections.

2.2 Google's paper and its main problem

In their paper [1], Google set out to compute the cross entropy fidelities for circuits with increasing qbit and gate count. They start off their discussion with a brief explanation on *crossentropy fidelity benchmarking* and on the physical implementation of their processor. Then, they explain how they generated their circuits and how they simulated them classically. All the results are then summarized in a single graph, in which they show that the computed fidelities are, within the margin of error, all greater than zero. But how did they compute them in the first place? Well, they actually executed and simulated three kinds of circuits: full random circuits and two simplified versions of those. The fidelities of the full circuits were computed only up to a point, after which it was not possible anymore. The simplified circuits' fidelities, however, were computed even for the circuit with the highest qbit and gate count. In the graph, they show that the fidelities for the three types of circuits match for every data point; thus, they expect that they would match even in the quantum supremacy regime (where they were able to compute only two out of the three).

Even though this last heuristic argument might still be considered valid, there is one bigger problem in Google's work: they actually did not perform the classical simulations with the best known algorithm. This was pointed out by IBM in [2], in which they proposed an (already known) algorithm based on tensor network which would have been able to simulate all of the circuits in the paper in less than a week. They did not bother to try and actually run the circuits, but their claims were confirmed about two years later, when in 2021 another independent group of researchers performed essentially the same quantum supremacy experiment but using IBM's proposed algorithm instead [3].

3 Reproducing the experiment

In this section, I want to see if and how it is possible to adapt Google's experiment to run on a different quantum processor; in particular, an IBM Eagle r3 [11]. I will highlight what were the crucial choices made by Google; then, I will show how I was able to adapt them to run a scaled-down version of the same experiment.

3.1 Google's methodology

[1] explains in great detail how the experiment was engineered and implemented physically. Here, I will restrict the discussion only to the crucial aspects of the circuits' implementation. More details are found in [10].

The Sycamore processor

Google specifically engineered their sycamore processor to tackle this task of proving quantum supremacy. The processor is made up of 53 superconducting qbits [12] arranged in a 2D lattice (Figure 2). The physical realization of the processor allows for the application of one-qbit gates and two-qbit gates across adjacent qbits. In their experiment, they used four different quantum gates: $\{\sqrt{X}, \sqrt{Y}, \sqrt{W}, fSim\}$; the latter two being defined as $W \doteq (X+Y)/\sqrt{2}$ and

$$fSim(\theta,\phi) \doteq \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -i\sin\theta & 0 \\ 0 & -i\sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & e^{-i\phi} \end{bmatrix}. \qquad (2)$$

The $fSim$ gate is used to create entanglement; its structure is dictated by the hardware implementation of the couplings between qbits. The parameters θ and ϕ can be controlled directly for each coupling, and they were tuned in order to minimize the gate noise (i.e. they were fixed for each coupling). Each $fSim$ gate was initialized with parameters $\theta \approx \pi/2, \phi \approx \pi/6$, which were then fed into an optimization loop to select the values that would result in the lowest possible error rates.

Ensuring complex circuits

The random circuits used in the experiment had to satisfy two constraints:

1) Sample efficiently from the Haar measure, and
2) be hard to simulate classically.

To satisfy (1), it was sufficient [6, 13] to prove that the chosen set of gates was universal (see [10]) and that the circuit depth m be great enough to ensure sampling from the Porter-Thomas distribution. In a previous paper [9], they argued that the depth should greater than the ballistic spread of entanglement across Hilbert space in chaotic systems (i.e. $m > n^{1/D} \approx 9$ layers).[2]

To satisfy (2), first Google made sure to include $\sqrt{W}$, which is not part of the Clifford set; thus guaranteeing that the circuits could not take advantage of the Gottesman–Knill theorem to be simulated easily [14, 5]. Second, they layered the gates in a way that, they claimed, would be hard to simulate by any known classical algorithm (Figure 3). This latter point turned out to be *flawed*, as different sources [2, 3] pointed out that there already existed an algorithm based on tensor network that could simulate this family of circuits in a few weeks' time.

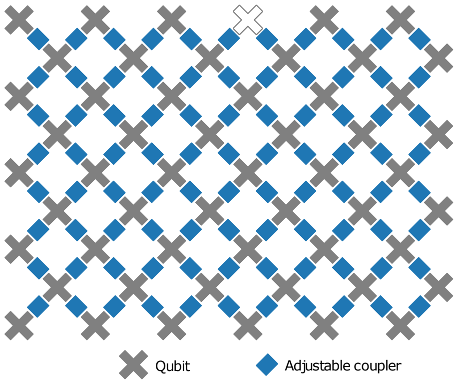


Figure 2: Sycamore processor used for Google's quantum supremacy experiment. The qbits are organized in a 2D lattice, with adjustable couplings in between. One qbit was broken and it was not used for the experiment.

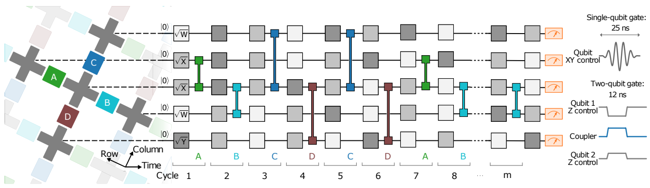


Figure 3: Example circuit from Google. The picture represents one layer for a subset of five qbits in one of their random circuits. Alternating sub-layers of gates and couplings ensure both randomness and computational hardness. No two same-type single-qbit gates are ever added after one another.

Running the circuits

Google prepared a series of quantum circuits with increasing depth and qbit count; they run them both in their Sycamore quantum computer and in a classical simulator. For each qbit and depth configuration, they prepared 10 different circuits running each one a number of times on the order of 10^6. They were able to simulate the full 53-qbit quantum circuits up to a depth of 14 layers by using a hybrid Schrödinger-Feynmann algorithm [15] running on their supercomputers. For deeper circuits, they claimed that the algorithms could not be run due to memory constraints and they resorted to using a simplified version of the circuits.

The quantum supremacy claims came from three observations:

1) They were not able to compute the full circuit fidelity after a certain threshold, due to the claims of the expected classical computing time being very large ($>$ 1e5 years).

2. $n \leq 53$ is the number of qbits and $D = 2$ is the dimensionality of the system.

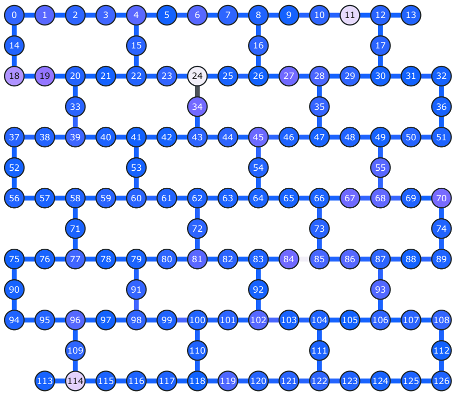


Figure 4: Qbit layout of "IBM Brismabe" Eagle r3 processor. The connections indicate the possible points in which the ECR gates can be applied. The qbits I used for the entanglement were: $0-8, 16, 26-32, 36, 51-45$

2) All the computed fidelities were significantly different from zero with an uncertainty margin of 5σ. This meant that the quantum computer was never completely overwhelmed by noise.
3) The fidelities of the full circuits and the simplified ones matched in every data point. Thus, they argued that the quantum computer was likely to operate correctly even for the deepest 53-qbit full circuits.

3.2 My methodology

The Eagle r3 processor

This family of processors is the only one currently made available to the general public. Even though the error rates are comparable to Sycamore, the qbit layout is different. In this case, entanglement cannot be easily created between an arbitrary set of qbits. To keep the number of quantum gates at a minimum, I entangled qbits in a chain, following a snake pattern (Figure 4). My choice of gates was constrained to $\{ECR, \sqrt{X}, X, RZ\}$, which are the ones physically implemented in the processor.[3] I did not do any noise-reduction optimization steps and I considered only the median error rates provided by IBM: $\sim$ 2.5e-4 for the $\sqrt{X}$ gate and of $\sim$ 7.9e-3 for the ECR gate[4]. The angle for the RZ gates was chosen at random, uniformly in $[-\pi, \pi]$. Given that the ECR gates are directional (not symmetric with respect to qbit swap), I only applied them in the direction that was physically implemented in the processor.

Ensuring complex circuits

To generate a circuit, I randomly chose gates from the set above, with uniform distribution, and add them at the back of the circuit on a random qbit. When an ECR gate is selected, it is added on two, randomly selected, adjacent qbits.

To satisfy claim (1) above, first I am assuming the former set of gates is universal[5]. Then, to select an adequate depth, I use the results of [6], which state that a circuit with $\mathcal{O}(n^2)$ entangling gates for should approximate well the first two moments of the Haar distribution for n qbits. Given that, on average, I place an entangling gate once every 4, I need to generate a total of $(4n)^2$ random gates to achieve just that.

Please, do not confuse this estimate with the one made for Google's circuit depth: in that case, increasing *depth* means adding a further layer which entangles *all* the qbits. In this computation, increasing *length* means adding just a single entanglement operation.

Satisfying claim (2) is not really a goal of mine. Google themselves did not manage to do that and there is no point for me in going out of my way to produce something that is supercomputer-proof. I am satisfied by my choice of non-Clifford gates, which means that my simulations are, at the very least, non-trivial.

Parameter	Value
Qbit count (n)	$10-25$, increments of 3
Gate count	$(4n)^2$
Number of circuits per data point	4
Number of runs per circuit	$10^4 - 10^5$, logarithmically spaced

Table 1: Some relevant parameters I used in the experiment

Running the circuits

I prepared a series of circuits with increasing length and qbit count, up to 25 qbits; I run them both in the Eagle r3 quantum computer and on a classical simulator.

For running the real circuits, I used IBM's `ibm_brisbane` quantum computer. The circuits were run as-is, without additional transpilation, and with an increasing number of shots. Funnily enough, I had no problems in simulating circuits up to 29 qbits, but *storing the simulated probabilities* past 25 qbits was impractical without some proper code structuring. The relevant simulations were done using Qiskit's `statevector` simulator [16] running on my $8GB$ GPU. I did not attempt to run any elided/path circuits.

3.3 My results

I would like to emphasize again that my goal was really just seeing what it would take to adapt the same experiment to a different architecture. By no means I was trying to claim supremacy of any kind, as this would require a very significant investment of time and resources. All of the parameters relevant to my experiment are summarized in Table 1.

According to classical simulations, the circuits' output effectively obeyed the Porter-Thomas distribution (Figure 5a). The same circuits, run on quantum hardware, yielded a disappointing result with a distribution being essentially uniform (Figure 5b). The fidelity computation confirmed that the noise was too high, with the value of zero being obtained across all circuits (Figure 6). There is one anomalous point in the last graph, but this turned out to be caused by a bug which disabled one of the couplings for 13-qbit circuits.

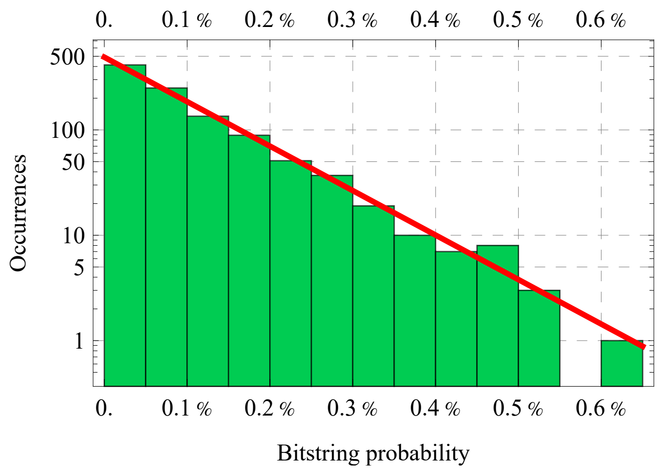
4 Conclusions

4.1 Experiment results

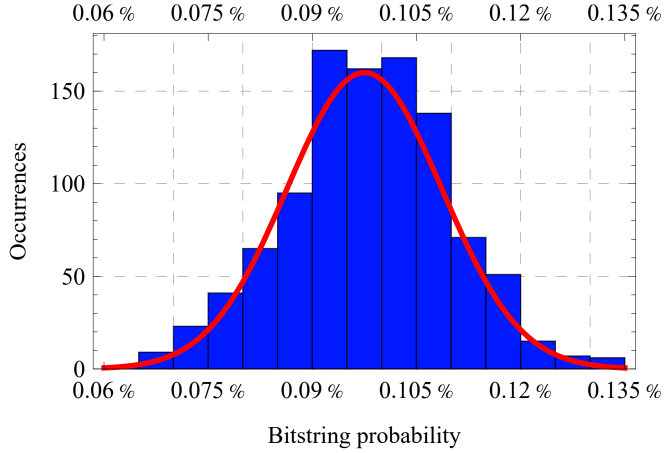
My poor results really show how much work Google has put into limiting the noise to a minimum. Even though the error rates are similar between the two processors, there are two main methodological differences that explain the discrepancy of the results: 1) the linear layout of the Eagle r3 processor needs more gates to obtain randomness, and 2) the tuning of entangling gates that Google performed was crucial to lowering the noise to an acceptable threshold.

3. $ECR \doteq (I \otimes X + X \otimes Y)/\sqrt{2}$
4. ECR is a 2-qbit maximally entangling quantum gate.

5. I could not find a reference with a proof of that. However, if it were not universal, they would most likely not have built a quantum computer around it.



(a) Probability distribution obtained from the simulation of a random 10-qbit circuit. The red line is the best linear fit of the data. We can clearly see that the distribution is exponential (plot is in logarighmic scale).



(b) Probability distribution obtained from 10^5 runs of the same 10-qbit circuit as (a). The red curve indicates the best gaussian fit of the data. The good fit indicates that the bitstrings are sampled uniformly due to the noise. Please note that I made this graph just to explain my results. In the actual experiment, the number of sampled points is usually too low for us to be able to see a proper distribution of probabilities.

Figure 5

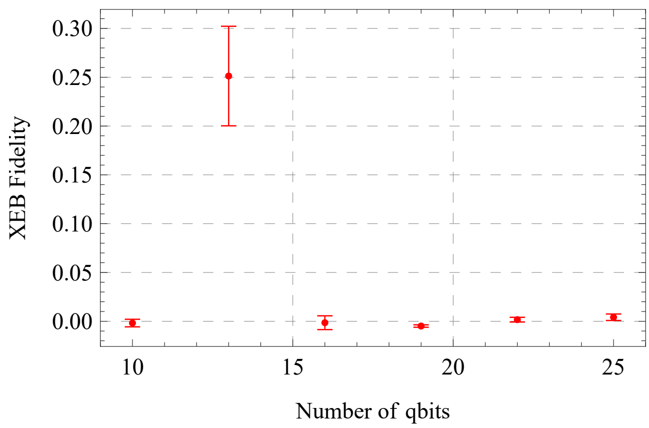


Figure 6: Linear cross-entropy benchmarking fidelity computations. All the data points are effectively zero, except for one anomalous result. This later turned out to be due to a bug which disabled one coupling for 13-qbit circuits. I left the data point in to show that it is, in fact, easy to spoof the results of this experiment.

4.2 Is XEB still relevant?

Even though it could have used some better wording and even though it had some flaws, Google's work was still impressive. Now, one could ask the question: where do we go from this? Is crossentropy benchmarking still the way to go to prove stronger forms of quantum supremacy? What transpires is that most researchers agree that XEB is going to become less and less relevant. The main problem is that if we increase the number of qbits even further, then the arguments for the validity of the (unverifiable) computations break down. In addition, there are studies that show that the linear crossentropy fidelity can be easily spoofed [17] (I did it by mistake in my results!). Still, there are many ways in which we can improve our supremacy claim. At the time of writing this, the latest claim came from D-Wave [18]. They used a task and hardware that are fundamentally different from XEB ; this shows that there are still many roads that we can follow in the pursuit of quantum supremacy.

References

[1] Frank Arute et al. "Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (Oct. 2019), pp. 505–510. ISSN: 1476-4687. DOI: 10.1038/s41586-019-1666-5. URL: http://dx.doi.org/10.1038/s41586-019-1666-5.

[2] Edwin Pednault et al. *Leveraging Secondary Storage to Simulate Deep 54-qubit Sycamore Circuits*. 2019. arXiv: 1910.09534 [quant-ph]. URL: https://arxiv.org/abs/1910.09534.

[3] Yulin Wu et al. "Strong Quantum Computational Advantage Using a Superconducting Quantum Processor". In: *Physical Review Letters* 127.18 (Oct. 2021). ISSN: 1079-7114. DOI: 10.1103/physrevlett.127.180501. URL: http://dx.doi.org/10.1103/PhysRevLett.127.180501.

[4] John Preskill. *Quantum computing and the entanglement frontier*. 2012. arXiv: 1203.5813 [quant-ph]. URL: https://arxiv.org/abs/1203.5813.

[5] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[6] Aram W. Harrow and Richard A. Low. "Random Quantum Circuits are Approximate 2-designs". In: *Communications in Mathematical Physics* 291.1 (July 2009), pp. 257–302. ISSN: 1432-0916. DOI: 10.1007/s00220-009-0873-6. URL: http://dx.doi.org/10.1007/s00220-009-0873-6.

[7] Joseph Emerson et al. "Pseudo-Random Unitary Operators for Quantum Information Processing". In: *Science* 302.5653 (2003), pp. 2098–2100. DOI: 10.1126/science.1090790. eprint: https://www.science.org/doi/pdf/10.1126/science.1090790. URL: https://www.science.org/doi/abs/10.1126/science.1090790.

[8] Alfred Haar. "Der Massbegriff in der Theorie der Kontinuierlichen Gruppen". In: *Annals of Mathematics* 34 (1933), p. 147. URL: https://api.semanticscholar.org/CorpusID:124917543.

[9] Sergio Boixo et al. "Characterizing quantum supremacy in near-term devices". In: *Nature Physics* 14.6 (Apr. 2018), pp. 595–600. ISSN: 1745-2481. DOI: 10.1038/s41567-018-0124-x. URL: http://dx.doi.org/10.1038/s41567-018-0124-x.

[10] Frank Arute et al. "Supplementary Information: Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (Oct. 2019), pp. 505–510. ISSN: 1476-4687. DOI: 10.1038/s41586-019-1666-5. URL: http://dx.doi.org/10.1038/s41586-019-1666-5.

[11] URL: https://docs.quantum.ibm.com/guides/processor-types.

[12] J. E. Mooij et al. "Josephson Persistent-Current Qubit". In: *Science* 285.5430 (1999), pp. 1036–1039. DOI: 10.1126/science.285.5430.1036. eprint: https://www.science.org/doi/pdf/10.1126/science.285.5430.1036. URL: https://www.science.org/doi/abs/10.1126/science.285.5430.1036.

[13] Joseph Emerson, Etera Livine, and Seth Lloyd. "Convergence conditions for random quantum circuits". In: *Physical Review A* 72.6 (Dec. 2005). ISSN: 1094-1622. DOI: 10.1103/physreva.72.060302. URL: http://dx.doi.org/10.1103/PhysRevA.72.060302.

[14] Daniel Gottesman. *The Heisenberg Representation of Quantum Computers*. 1998. arXiv: quant-ph/9807006 [quant-ph]. URL: https://arxiv.org/abs/quant-ph/9807006.

[15] Ethan Bernstein and Umesh Vazirani. "Quantum Complexity Theory". In: *SIAM Journal on Computing* 26.5 (1997), pp. 1411–1473. DOI: 10.1137/S0097539796300921. eprint: https://doi.org/10.1137/S0097539796300921. URL: https://doi.org/10.1137/S0097539796300921.

[16] Ali Javadi-Abhari et al. *Quantum computing with Qiskit*. 2024. DOI: 10.48550/arXiv.2405.08810. arXiv: 2405.08810 [quant-ph].

[17] Boaz Barak, Chi-Ning Chou, and Xun Gao. *Spoofing Linear Cross-Entropy Benchmarking in Shallow Quantum Circuits*. 2020. arXiv: 2005.02421 [quant-ph]. URL: https://arxiv.org/abs/2005.02421.

[18] Andrew D. King et al. *Computational supremacy in quantum simulation*. 2024. arXiv: 2403.00910 [quant-ph]. URL: https://arxiv.org/abs/2403.00910.