Quantum Hardware Roofline: Evaluating the Impact of Gate Expressivity on Quantum Processor Design

Justin Kalloor[1], Mathias Weiden[1], Ed Younis[2], John Kubiatowicz[1], Bert De Jong[2], and Costin Iancu[2]

[1]Department of Electrical Engineering and Computer Science, University of California, Berkeley
{*jkalloor3, mtweiden, kubitron*}*@cs.berkeley.edu*
[2]Computational Research Division, Lawrence Berkeley National Laboratory
{*edyounis, wadejong, cciancu*}*@lbl.gov*

2024 IEEE International Conference on Quantum Computing and Engineering (QCE) | 979-8-3315-4137-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/QCE60285.2024.00100

***Abstract*— The design space of current quantum computers is expansive, with no obvious winning solution, leaving practitioners with a crucial question: "What is the optimal system configuration to run an algorithm?" This paper explores hardware design trade-offs across NISQ systems to better guide algorithm and hardware development. Algorithmic workloads and fidelity models drive the evaluation to appropriately capture architectural features such as gate expressivity, fidelity, and crosstalk. As a result of our analysis, we extend the criteria for gate design and selection from only maximizing average fidelity to a more comprehensive approach that additionally considers expressivity with respect to algorithm structures. A custom synthesis-driven compilation workflow that produces minimal circuit representations for a given system configuration drives our methodology and allows us to analyze any gate set effectively. In this work, we focus on native entangling gates (CNOT, ECR, CZ, ZZ, XX, Sycamore, $\sqrt{\text{iSWAP}}$), proposed gates (B Gate, $\sqrt[4]{\text{CNOT}}$, $\sqrt[8]{\text{CNOT}}$), as well as parameterized gates (FSim, XY). By providing a method to evaluate the suitability of algorithms for hardware platforms, this work emphasizes the importance of hardware-software codesign for quantum computing.**

I. Introduction

Quantum computers offer an exciting opportunity to explore problems previously considered intractable. An assortment of companies have introduced gate-based quantum machines that range in qubit technology: superconducting transmon qubits [4], fluxonium qubits [3], trapped-ion qubits [10], neutral atoms [20], and several others.

Practical questions have already arisen in the community related to the comparison of different hardware solutions: "What computer should I use to run my algorithm? How can I improve my current quantum processor? What gates should I provide to end-users?"

In the current Noisy Intermediate Scale Quantum (NISQ) computing era, the most important performance metric for current systems is their ability to execute algorithms with the least amount of error, i.e. maximize algorithmic fidelity.

To this end, hardware designers attempt to improve the accuracy of an algorithm's execution using a multi-stage design process aimed at optimizing behavior across multiple hardware characterization criteria: gate fidelity, crosstalk (gate parallelism and qubit connectivity), etc. This process, centered around gate fidelity, proceeds as follows: First choose a native entangling two-qubit gate that can be implemented with high fidelity that is "good enough" to represent any two qubit process (unitary). After this, develop additional techniques, e.g. crosstalk mitigation, to further improve gate fidelity.

We believe that this design process can be improved upon from both a hardware and end-user perspective. Most hardware characterization metrics are algorithm agnostic [12], [21], [26]; therefore, these widely accepted metrics (e.g. gate fidelity) are hard to correlate directly with algorithm performance across systems with distinct hardware characteristics. Full algorithm fidelity models that capture hardware characteristics (gate fidelity and parallelism/crosstalk) have been introduced in the literature [9]. While they are able to assess the fidelity of an algorithm when executed on a single hardware configuration, these models still lack predictive power when varying architectural parameters. The problem stems from the fact that these metrics combine algorithm-agnostic hardware characterization metrics with metrics that characterize the program implementation and resource consumption (e.g. gate count, circuit depth), and implicitly the impact of the program generators and compilers.

In this paper we argue that gate set design should be driven by representational power in the context of a given algorithm or algorithmic workload. In order to attain the most resource efficient implementation, we use custom compilation workflows that combine traditional compilers, such as Cirq [14] or Tket [40], with circuit synthesis tools [45].

This paper makes the following contributions:

First, we leverage analytical models that combine hardware (gate fidelity, parallelism and qubit connectivity) and algorithm implementation (gate count, depth) characteristics to construct a comparative performance *roofline* for hardware and compiler designers, as well as system end-users. Our quantum roofline is able to derive which particular metric can lead to overall improvements in algorithmic fidelity, as well as upper bounds on these metrics past which no additional end-user gains can be expected. For example, when comparing Sycamore (Google) and CNOT (IBM) entangling gates for a particular algorithmic workload, our analyses show that there are ranges of relative gate fidelities where one configuration can always outperform the other! Once a certain *threshold* fidelity has been attained, no improvements in one- or two-

979-8-3315-4137-8/24/$31.00 ©2024 IEEE
DOI 10.1109/QCE60285.2024.00100

Hardware	# Qubits	Technology	Connectivity	Gate Set	1Q/2Q Error(RB/XEB)
Google Sycamore	54	Superconducting	Mesh	1Q: XZ, RZ 2Q:FSim,$\sqrt{\text{iSWAP}}$, Sycamore, CZ	0.001 / 0.01
IBM Eagle r3	127	Superconducting	Mesh	1Q: SX, RZ, X 2Q: ECR	0.0002 / 0.007
IBM Hummingbird r3	65	Superconducting	Mesh	1Q: RZ, SX, X 2Q: CNOT	0.00027/ 0.012
IBM Falcon r5	27	Superconducting	Mesh	1Q: RZ, SX, X 2Q: CNOT	0.0003/ 0.0079
Rigetti Aspen M3	79	Superconducting	Mesh	1Q: RX, RZ 2q: CZ, XY	0.001/ 0.14, 0.092
Quantinuum H2	32	Ion Trap	A2A	1Q: U1, RZ 2Q: ZZ	3e-5/0.002
IonQ Forte (2024)	35	Ion Trap	A2A	1Q: GPi, 2Q: ZZ, XX	0.0002/ 0.0040
IonQ Aria	21	Ion Trap	A2A	1Q: GPi, 2Q: XX	0.0006/ 0.0040

TABLE I: *Summary of existing commercial Quantum Computing hardware [2], [14], [19], [34], [37]. As we can see, most devices available now are superconducting or ion trap devices, with superconducting devices proving to be easier to scale. This contrasts the ion trap devices which show on average higher RB fidelity. Additionally, superconducting qubits have a mesh (2D Nearest Neighbors) topology while ion traps are all-to-all (A2A).*

qubit gate fidelity on any architecture can lead to better relative performance with respect to the other.

Next, we introduce a circuit synthesis based compilation procedure which indicates that the existing gate set design criteria that favors choosing gates based on their attainable fidelity and representational power of random two-qubit process may be misleading. Instead, our analysis shows that the criteria should be augmented with their representational power for multi-qubit processes (e.g. three qubit sub-unitaries appearing within larger circuits) that are drawn from implementations of existing algorithms. Previous work [42] illustrates that the the span of unitaries appearing in algorithms is much smaller than the space of all possible unitaries. As our compilation tool-chain synthesizes these unitaries, rather than the given quantum circuit, we can evaluate a gate's ability to represent an algorithm. For example, while the B-gate [46] is the most expressive gate for two qubit unitaries, we cannot uncover advantages when using it to represent complex programs. When compared against CNOT, B-gates surprisingly lead to gate count increases and possible fidelity decreases. At the other end of the spectrum, we show that several low-entanglement gates such as $\sqrt[4]{\text{CNOT}}$ and $\sqrt{\text{iSWAP}}$ are sometimes able to offer similar expressive performance as maximally entangling gates for important circuits such as TFIM, QFT, and QAE, leading to better algorithmic fidelity.

As discussed in Section X, we believe our assessment procedure extends well beyond NISQ into the Fault Tolerant era of quantum computing.

II. Quantum Hardware Characterization and Benchmarking

Today's systems are dominated by superconducting (IBM, Google) and trapped ion (Quantinuum, IonQ) qubits. Neutral Atom (QuEra, Atom Computing) and silicon-spin qubits (Intel) are starting to gain traction, while several other technologies are continuously being developed. All these systems expose to end-users a universal gate set [31], composed of single-qubit and entangling two-qubit native gates. These gate-set choices are outlined in Table I. Several methods exist to characterize today's quantum machines:

Average Gate Fidelity: The widest used characterization and processor optimization metric is the *average gate fidelity*, which captures the probability that a state does not succumb to any error when a gate is applied. Fidelity can be measured using Randomized Benchmarking (RB) protocols [21], [26], [27], which use random Clifford gates to create a depolarization channel over a set of qubits with a single probability p, the average infidelity of the gate's application. By using variable lengths of random Clifford circuits, existing protocols calculate the infidelity per gate (p_{gate}): fidelity is then computed as $1 - p_{\text{gate}}$.

In 2019, Google introduced the cross entropy benchmarking (XEB) protocol [1] as another way to compute average gate fidelity. Importantly, this study also shows that the total average fidelity of a circuit can be approximated using a simple *digital error model*, validated for NISQ size systems.

Process Fidelity: As RB protocols have trouble scaling past three qubit processes, Cycle Benchmarking (CB) [15] has been proposed to improve characterization scalability to larger processes (and hardware). CB based protocols indicate that besides average gate fidelity, hardware dependent metrics such as qubit connectivity and algorithm specific metrics such as gate parallelism per cycle need to be taken into account when assessing algorithm fidelity.

Quantum Volume: Quantum Volume (QV) [12] characterizes the capability of hardware to execute random circuits of a certain size. This metric cannot be used to compare the fidelity of different process implementations running on the same machine or that of a single process running across different machines.

Algorithmic Qubits: IonQ's Algorithmic Qubits (AQ) metric [9] captures a system's ability to execute an algorithmic workload. AQ protocols measure the largest number of effectively perfect qubits you can deploy for a typical quantum program. It is similar to QV, but it additionally considers quantum error correction and presents a clear and direct relationship to qubit count. The AQ metric captures the impact of the compilation tool-chain.

All of these protocols and metrics reveal different useful information about a single configuration of quantum machine. However, they all fail when comparing across different hardware and gate sets. While we can measure the fidelity and quantum volume of a CNOT-based machine and of a Sycamore-based machine, this does not give us any information on their respective abilities to run a given algorithm. AQ encapsulates the algorithmic potential of different hardware, but it is still unable to quantify the degree to which one needs make changes to an architecture's configuration in order to provide better comparative performance.

In order to make these inferences, we advocate for an

algorithmic workload based approach centered around circuit fidelity models that combine hardware characterization metrics with the hardware's ability to represent and implement a particular algorithm. We start with the simple digital model based on average gate fidelity, which we then extend to account for crosstalk due to parallel gate execution as well as qubit connectivity (an idle qubit can be affected when executing an operation on a neighboring qubits).

A. Roofline

The *roofline* model [43] provides an intuitive understanding of performance bottlenecks in classical hardware. The model outputs the maximum attainable performance on a machine as a function of the operational intensity of an application. This allows hardware and software architects to understand the inherent limits of their design in the context of important universal metrics: computational power and network bandwidth.

Our paper introduces a quantum analog, which highlights bottlenecks in algorithmic fidelity when comparing two different machines by gate expressivity, fidelity, and connectivity. While the classical roofline is fixed for a particular hardware, our relative roofline is fixed for an algorithmic workload. This allows for simple comparisons to be made between quantum computers for targeted domains. As a consequence, our model is compiler dependent, which motivates our custom compiler workflow and the importance of numerical synthesis based transpilation (Section III-B).

III. Quantum Algorithms and Compilers

The digital model indicates that circuit fidelity is determined by the average gate fidelity and the circuit gate count: improving both metrics will improve algorithmic fidelity. The gate count for a given algorithm implementation is determined by hardware characteristics: 1) representational power of the native gate set; and 2) qubit interconnection topology.

Due to exponentially compounding gate infidelities, the dominant factor in the digital model is gate count. This has two consequences: 1) comparisons between system configurations should be done using the implementation with the fewest number of gates attainable, together with the lowest depth or highest gate parallelism; and 2) the compilation toolchain plays a very important role in determining the overall "performance" of a given configuration.

A. Quantum Algorithms

Domain generators [29], [34] that produce the circuit associated with a given algorithm tend to have the following common characteristics:

- They are developed to generate circuits in a restricted gate set. Most generators use directly the CNOT gate, while some hardware-vendor-provided generators target only vendor supported native gates.
- The generated circuits have a logical qubit connectivity that resembles the domain level structure. For example, optimal QFT circuits are generated assuming an all-to-all qubit connectivity. Circuits generated for fermionic [29] interactions map fermions to qubits using a logical topology that resembles the structure of the physical system modeled.
- Some generators are deemed optimal. Optimality here relates only to to asymptotic complexity: a good compiler can greatly reduce the constants that appear in the complexity formula.

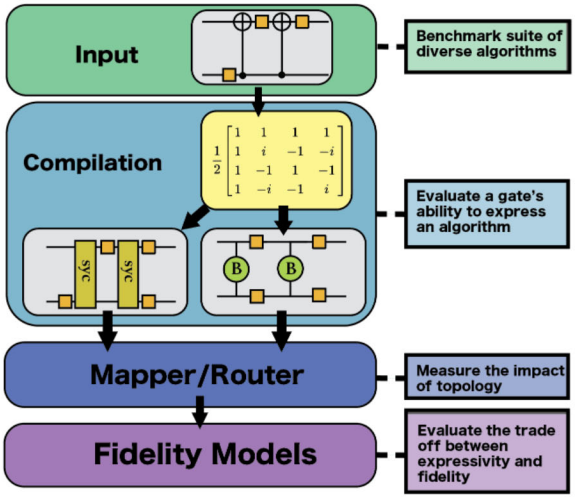

Fig. 1: *Our Hardware Comparison Procedure: A synthesis-based cross-compilation process (sometimes called "transpilation") allows us to explore multiple gate sets and the ability to express an algorithm in terms of gate count, depth, and parallelism. From there, we can understand the effects of topology and fidelity on the overall performance of a quantum machine.*

Due to these properties, a quantum compiler's ability to: 1) eliminate gates that are redundant or can be simplified from the circuit; 2) map and route the input circuit to the hardware configuration; and 3) translate (transpile) the input circuit to a different gate set is paramount for accurate architectural comparisons.

B. Quantum Compilers

Traditional vendor compilers use peephole optimizations based on 2-qubit gate synthesis (KAK decomposition [41]), application of gate commutativity rules, or domain specific pattern rewriting rules (*e.g.* Tket's phase gadgets [11]). They also provide mapping and routing algorithms [22], [34], [40] and translation between multiple gate sets ("transpilation"). In particular, transpilation is performed using a 1:1 gate rewriting rule: any 2-qubit gate is rewritten directly from one (e.g. CNOT) to another (e.g. Sycamore).

Circuit synthesis [36], [45] based tools have been introduced recently and have been shown to provide better quality implementations [13], [25], [42], [44] when compared against vendor compilers, albeit at the expense of increased compilation time. Some of these tools integrate optimization [13] with mapping [25] and gate transpilation [44]. They can search a large space of circuit structures and transformations.

All compilation tools have one thing in common: the compilation workflow is custom and it consists of repeated applications of passes and transformations. While it is hard to quantify the impact of optimization, mapping, and transpila-

Family	Benchmarks (circuit_width)
TFIM	TFIM_16, TFIM_64, TFXY_16, TFXY_64
QAE	qae_9, qae_13, qae_17, qae_21
QFT	qft_4, qft_12, qft_64
QAOA	qaoa_10
QPE	qpe_14, qpe_18
Adder	adder_9, adder_63, mul_10, mult_60
Shor	shor_12, shor_16, shor_24, shor_28
Grover	grover_5
Hubbard	hubbard_4, hubbard_8, hubbard_12
QML	qml_6, qml_13, qml_22
VQA	vqe_12 (LiH), vqe_14 (BeH_2)

TABLE II: *Our Benchmarks: List of benchmark circuits organized by family. Each circuit was initially created with CNOT and U3 gates.*

Benchmark	Size	Gate	BQSKit		Tket		Cirq		Qiskit	
			a2a	mesh	a2a	mesh	a2a	mesh	a2a	mesh
mul_10	163	cz	**67**	**89**	104	134	134	227	132	141
		b	**110**	**125**	208	480	-	-	-	-
		syc	**103**	**139**	208	319	260	364	-	-
qft_16	264	cz	**237**	**336**	240	522	264	563	264	426
		b	**242**	**302**	480	1044	-	-	-	-
		syc	**241**	**365**	480	828	288	755	-	-
TFIM_16	240	cz	**200**	**200**	240	240	240	240	240	240
		b	**200**	**202**	480	480	-	-	-	-
		syc	**200**	**208**	480	219	440	440	-	-

TABLE III: *Comparison of two qubit gate counts for a subset of the benchmarks and gate sets across our different compilers. The first number under each compiler is for an all-to-all topology system, while the second number is for a mesh topology. Synthesis based compilers, such as BQSKit, produce the circuits with the least amount of gates. Note that Cirq (Qiskit) is unable to compile to the B Gate (B and Sycamore Gate).*

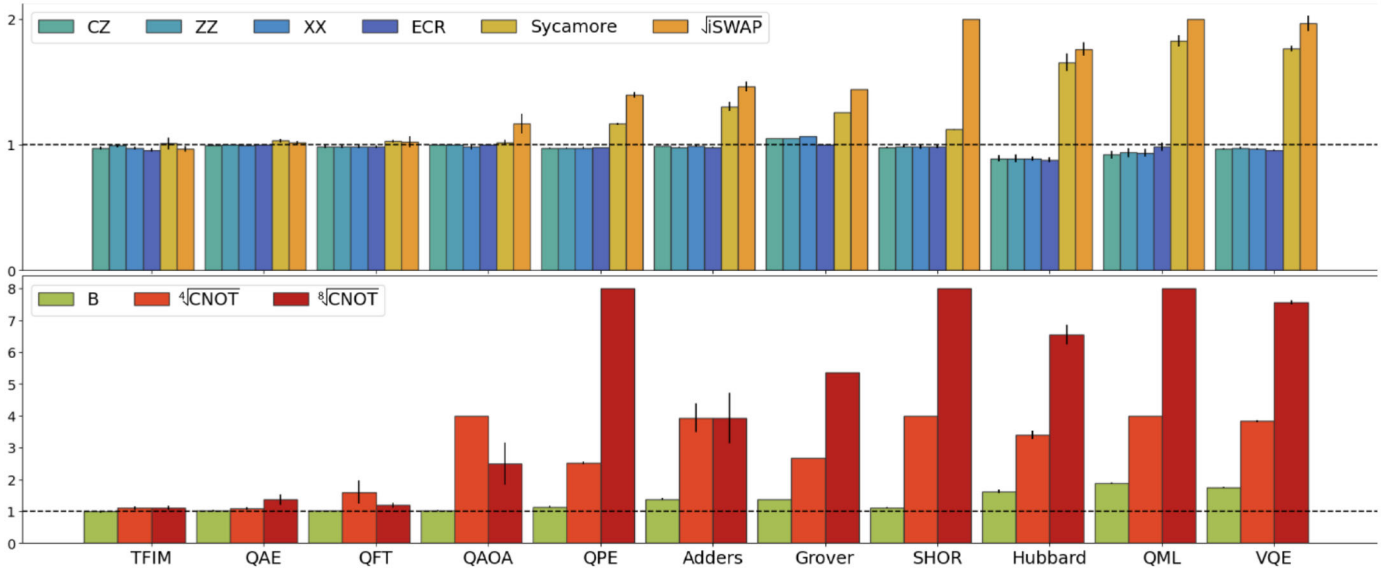

Fig. 2: *Normalized two-qubit gate count for existing hardware (top) and theorized hardware (bottom). We plot the relative count with respect to the optimally compiled CNOT based circuit. The gate count encapsulates logical algorithm connectivity and serves as a proxy for a gate's ability to represent the algorithm.*

tion phases in isolation, we note that compilers can realize up to an order of magnitude in gate count reduction, even when the input circuit is "optimally" generated.

IV. Evaluation Procedure

We used the algorithms shown in Table II for our evaluation. They include many important categories including Variational Algorithms (VQA and QAOA) [16], [33], Finance (QAE) [17], Number Theory (QFT, QPE, Shor) [5], Physical Simulation (Hubbard, Ising(TFIM)) [7], [39], Search (Grover [28]), and Quantum Machine Learning (QML). The QML circuit is based on [8] and has an n-bit encoder and a two-local network. Most benchmarks were generated using Qiskit circuit generators [34], while the TFIM circuits were generated with F3C++ compiler [32]. For each algorithm we generate several instances across inputs and circuit sizes (number of qubits). Overall, we believe that our algorithmic workload provides a good sampling of the space of circuit implementations. We consider up to 64 qubit programs, with gate counts as high as 37000 accounting for a maximum depth of 44500. The logical topology of these programs ranges from linear in TFIM to all-to-all in QFT.

We translate and optimize the benchmarks for native gates present in today's hardware (CNOT, ECR, CZ, ZZ, XX, Sycamore, $\sqrt{\text{iSWAP}}$), as shown in Table I. Additionally, we examine experimental gates theorized to provide algorithmic fidelity advantages due to either high expressivity or high fidelity, B and $\sqrt[4]{\text{CNOT}}$ or $\sqrt[8]{\text{CNOT}}$ respectively.

An overview of our process is shown in Figure 1. We use a custom compilation workflow that first performs rule-based and algorithm-level transformations (Tket, Qiskit, Cirq) and pass these optimized circuits to BQSKit for transpilation and mapping ([25], [44]). We perform a search across all of these compilers to select the best resultant circuits. Table III shows a sampling of these results. As described in [44], numerical synthesis partitions a larger circuit into three qubit blocks. By considering the underlying unitary of each block rather than the reference circuit, the synthesis algorithm is able to perform a global optimization over the entire block. This provides a sizeable advantage when compared to a standard rule-based gate translation, which only considers the two-qubit interaction of the original gate. To our knowledge, this workflow generates the best attainable implementations of a given algorithm on a given hardware configuration. These

resource "optimal" circuits[1] are then run through our fidelity models described in Section VI in order to compare overall performance.

V. Gate Representational Power

Given a reference implementation using CNOT gates, in Figure 2 we show the relative two-qubit gate count (averaged across all circuits in a circuit family) after re-targeting to a particular gate. This data shows the ability of a gate to represent a particular algorithm, i.e. it captures its expressivity and entanglement power. When considering existing native gates, the {CZ, ZZ, XX, ECR} set seems to have the same representational power as CNOT gates. The native gates {Sycamore, $\sqrt{\text{iSWAP}}$} have lower representational power than CNOT, as illustrated by their higher gate counts. When considering theorized gates, we see that {B, $\sqrt[4]{\text{CNOT}}$, $\sqrt[8]{\text{CNOT}}$} have overall lower representational power than CNOT. This is surprising as the motivation behind the introduction of the B-gate was its optimal 2-qubit representational power.

This behavior is also algorithm dependent. The TFIM, QAE, and QFT circuits require almost the same number of gates, irrespective of which gate it is used. For the rest of the circuits, we see more nuanced behavior. The gate set {CZ, ZZ, XX, ECR, CNOT} leads to the least amount of gates used, with much higher gate counts for the set {Sycamore, $\sqrt{\text{iSWAP}}$, B, $\sqrt[4]{\text{CNOT}}$, $\sqrt[8]{\text{CNOT}}$ }.

This data indicates that the machine with the highest gate fidelity will be best suited to execute TFIM, QFT, and QAE. For the rest of the algorithms, the best machine will consider trade offs in both gate fidelity and circuit structure (gate count, parallelism etc.). Our data also indicates that gate representational power with respect to full algorithms needs to be taken into account when selecting a system configuration. We discuss in detail representational power trade-offs in Section IX.

VI. Circuit Fidelity Models

In the NISQ era, it is critical to maximize the probability that a circuit's output state is correct. The output expectation can be described a function of the average gate fidelity of the machine, written as [31]:

$$\mathbf{F}_{\text{gate}}(\mathcal{E}, U) = \int d\psi \, \langle\psi| \, U^\dagger \mathcal{E}(\psi) U \, |\psi\rangle$$

where U is the target unitary and $\mathcal{E}$ is the erroneous channel trying to implement U.

To assess algorithm fidelity on a particular system configuration, we use a series of models that capture circuit characteristics together with an increasing number of architectural features: (1) gate fidelity; (2) gate fidelity and parallelism. In Section VII-E we discuss a model based on qubit connectivity as well.

Let $\mathbf{F}(\cdot)$ denote a circuit fidelity model, and let A and B denote two distinct system configurations. In order to enable system comparisons, we analyze the *objective function* given by:

$$\pi = \mathbf{F}^A(\cdot) - \mathbf{F}^B(\cdot)$$

[1]Meaning they do not improve with further compilation and optimization.

A. Gate Fidelity

Our first model is derived from [1], in which the authors verify that the measured fidelity and estimated fidelity based on this model track almost exactly for their tested circuits.

*Definition 1 (***Digital Fidelity Model):** The average circuit fidelity $\mathbf{F}_d$ can be estimated as

$$\mathbf{F_d} = \prod_{i=1,2,..} f_i^{n_i}$$

where n_i is the number of i-qubit gates in the circuit, and f_i is the average fidelity of an i-qubit gate. For systems with only one- and two-qubit native gates this becomes:

$$\mathbf{F_d} = f_1^{n_1} \cdot f_2^{n_2}$$

with the objective function:

$$\pi_{\mathbf{d}} = {f_1^A}^{n_1^A} \cdot {f_2^A}^{n_2^A} - {f_1^B}^{n_1^B} \cdot {f_2^B}^{n_2^B}$$

B. Gate Fidelity and Parallelism

In many systems, parallel execution impacts the attainable gate average fidelity. To capture this, we use a model based on Cyclic Benchmarking [15]. The protocol considers circuits as a series of cycles, with which we can calculate a single cycle fidelity as function of the 1-qubit and 2-qubit process fidelities (γ). The process fidelity and average fidelity as defined above are related by a simple linear equation [18].

*Definition 2 (***Cyclic Fidelity Model):**

$$\mathbf{F_c} = \prod (1 - e_i \cdot P_i)^m$$

where P_i is the average parallelism of i-qubit gates in the circuit, m is the depth of the circuit, and e_i is the average process infidelity for an i qubit gate $(1-\gamma_i)$. e_i can be measured using the Cycle Benchmarking protocol. The objective function for our machine comparison becomes:

$$\begin{aligned}\pi_{\mathbf{c}} &= {\mathbf{F_c}}^A - {\mathbf{F_c}}^B \\ &= (1 - e_1^A \cdot P_1^A)^{m^A} \cdot (1 - e_2^A \cdot P_2^A)^{m^A} \\ &- (1 - e_1^B \cdot P_1^B)^{m^B} \cdot (1 - e_2^B \cdot P_2^B)^{m^B}\end{aligned}$$

VII. Evaluating Quantum Machines

Now, we can finally answer the question "*What machine should I use to run my algorithm?*".

While the hardware landscape is constantly changing, by today's numbers Quantinuum boasts by far the highest 2-qubit gate fidelity at 0.998 and its ZZ gate can express our algorithmic workload well. For algorithms that use more qubits than H2's capacity, the models suggest the IBM Eagle system.

A more insightful question is how could system configurations be changed in order to improve competitiveness, e.g. : *"How can other machines become better than H2?"*.

A. Quantifying Design Trade-offs

Our procedure allows us to quantify the trade-offs between a gate's representational power for an algorithm and its fidelity. This is a comparative analysis where we vary the models' parameters and solve for the objective function as defined in Section VI. As gates continue to improve and

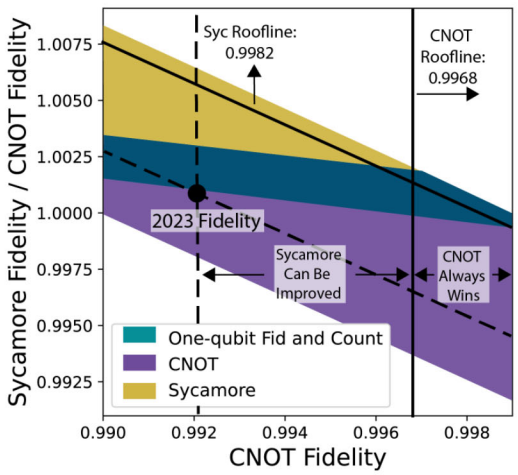

Fig. 3: *Machine capability to execute the adder_9 algorithm. CNOT fidelity is on the x-axis and the relative Sycamore fidelity on the y-axis. We plot the winning machine at each point. The middle area shows where the choice of best machine is a function of the single-qubit fidelity. In the other areas, each machine wins irrespective of 1-qubit gate fidelity. The published 2-qubit gate fidelities are shown with the black dotted lines, while the corresponding rooflines are plotted with solid black lines. The CNOT (Sycamore) gate always wins when its fidelity is right of (above) the roofline.*

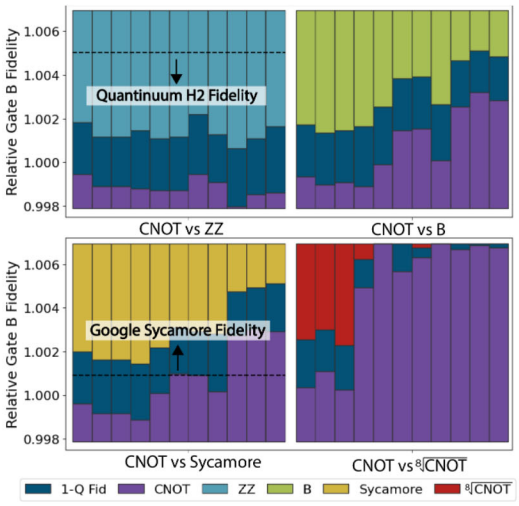

Fig. 4: *Gate set comparison against CNOT. CNOT fidelity is fixed to the IBM Falcon (vertical dotted line in Figure 3). The bars correspond to a vertical slice of the full analysis (Figure 3) for each of the circuit families: TFIM, QAE, QFT, QAOA, QPE, Adders, Grover, Shor, Hubbard, QML, and VQE. Encouragingly, low entanglement gates ($\sqrt[8]{CNOT}$) can provide better overall circuit fidelity for several important algorithms.*

calibration/noise-mitigation techniques advance, architects and end-users must consider:

1) "What gate fidelity do I need in order to out-perform other machines? How does this vary by algorithm class?"
2) "Does single-qubit gate count matter for relative performance?
3) "Is offering multiple entangling gates worth the development and maintenance effort?"
4) "How does the underlying chip topology affect the relative performance?"

Given that the maximum attainable gate fidelity is 1, in order to compare NISQ-era devices, we want to use realistic constraints. First, to simplify the model, we will initially limit the single-qubit gate type to only the $U3$ gate. Current machines have parameterizable rotation gates that can be composed to perform any arbitrary single-qubit unitary. As 2-qubit gate errors still dominate, this simplification is justified. Based on Table I, single-qubit gate fidelities vary from around 0.999 to 0.99999 across all quantum machines considered, while 2-qubit gates range from 0.990 to 0.999.

B. Two-Qubit Gate Analysis

We use the *adder_9* algorithm to directly compare IBM Falcon (CNOT) with Google Sycamore (Sycamore) machines as our driving example. The corresponding objective function (defined in Section VI) is:

$$\pi_{\mathbf{d}} = {f_1^A}^{70} \cdot {f_2^A}^{49} - {f_1^B}^{91} \cdot {f_2^B}^{66}$$

For comparisons, we rewrite the objective function to use the Sycamore fidelity relative to CNOT. We vary the CNOT fidelity along the x-axis and the relative Sycamore fidelity along the y-axis. This leaves the single qubit fidelities (f_1^A, f_1^B) free:

$$\pi_{\mathbf{d}} = {f_1^A}^{70} \cdot x^{49} - {f_1^B}^{91} \cdot (x \cdot y)^{66}$$

Each point represents a two-system configuration we are comparing, with the 2-qubit fidelities set according to the x and y position. The fidelities are bounded by our model constraints, resulting in the colored band as shown in Figure 3. We identify three behavioral regions. In the two outer regions, one configuration wins against the other *no matter the single-qubit gate fidelity of the system*. In these regions 2-qubit gate fidelity and expressivity fully determine the relative system behavior. In the central region, the winning machines depends on the fidelity of single-qubit gates (ratio between f_1^A and f_1^B): one machine can be improved relative to the other by tuning their single-qubit gate fidelity. For each system we also compute a *2-qubit gate threshold fidelity*, shown with continuous lines: once that is reached on a system, no improvements[2] in the other system's 2-qubit gate fidelity will change the overall ordering. We also plot the published fidelities of the respective hardware gates with a dotted line. The distance between actual and threshold fidelity for a gate indicates a window of opportunity to improve the other system.

We refer to this method of relative comparison as a *quantum hardware roofline*, as it allows us to compute bounds on the required improvements for a particular system configuration. For example, In Figure 3, once the CNOT gate reaches the *threshold* fidelity of 0.9968, no improvements in the Sycamore fidelity will outperform a CNOT based machine.

Figures 4 and 5 extend these results across algorithm classes and gatesets. Figure 4 compares several gates against the CNOT gate whose fidelity is fixed to that of the IBM Falcon system (each bar is a vertical slice of the plot in Figure 3 at the IBM Falcon fidelity). Again, configurations

[2]We vary the fidelities within the constraints of our model.

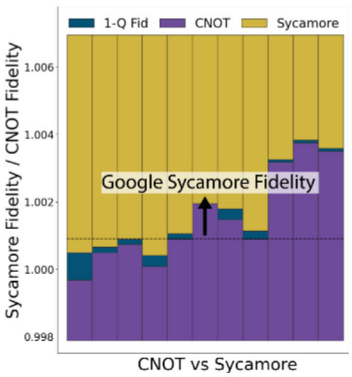

Fig. 5: *Sycamore Fidelity Comparison across algorithm classes using the cyclic fidelity model [15]. We see similar trends in terms of machine-dominant regions and rooflines, however the single-qubit impact is much smaller with this model.*

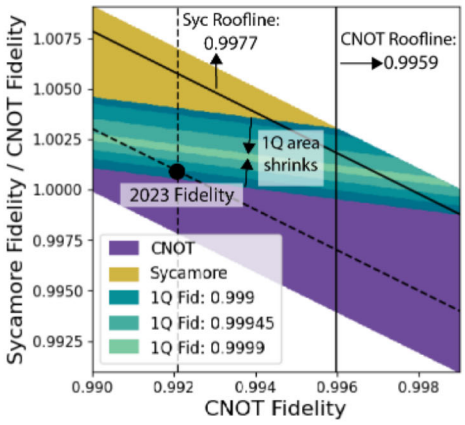

Fig. 6: *This plot shows the behavior of the 1-qubit dependent region as we fix the 1-qubit fidelity. The 1-qubit region shrinks as the fidelity increases, and disappears entirely when you reach a 1-qubit fidelity of 0.999988 for adder_9.*

Machine 1	Machine 2	Thresh
IBM Falcon	IBM Eagle	1.46
IBM Falcon	Sycamore	1.70
IBM Falcon	H2	1.06
IBM Eagle	Sycamore	2.69
IBM Eagle	H2	1.69
Sycamore	H2	1

TABLE IV: *The upper bound on 2-qubit gates ratio that determines the impact of 1-qubit gate tuning for selected pairs of NISQ machines. We use gate fidelity provided by the data-sheets for each machine [14], [34], [35]. Gate count ratios less than the threshold ratio signify circuits where tuning 1-qubit gates can improve the relative performance. For Sycamore vs. H2 (1), the only way to improve relative performance is by improving 2-qubit gate fidelity, while for the Falcon vs. H2 (1.06) there is a slight window of opportunity.*

can be improved by improving only 2-qubit fidelity, or by improving 1-qubit and 2-qubit gate fidelity together. The exact behavior is algorithm and gate set dependent. Encouragingly, low entanglement gates can provide advantages for some algorithms. Figure 5 shows this behavior when targeting the Cyclic Model. The trends are similar across models, with the Cyclic Model placing a much smaller emphasis on the single-qubit configuration. This is to be expected since the model emphasizes the impact of 2-qubit gates.

C. Single-Qubit Gate Analysis

To understand the implications for 1-qubit gate design we consider the closure of the 1-qubit dependent behavior (middle region) across "any" algorithm. To this end, we constrain the 1-qubit gate count as a function of the two-qubit gate count. We lower bound the 1-qubit gate count as $\frac{1}{8}$ of the corresponding 2-qubit gate count, far below the ratio found in any of our experimental data. We upper bound with twice the 2-qubit gate count: any consecutive U3 gates can be combined into one, so there are at most two 1-qubit gates in between the 2-qubit gates.

Now, we can vary the 1-qubit gate count between these bounds (keep n_1^A and n_1^B as variables). Then, we can analyze how the objective function behavior changes as we fix single-qubit fidelity (F_1) for both machines.

$$\pi = (F_1)^{n_1^A} \cdot x^{4153} - (F_1)^{n_1^B} \cdot y^{5944}$$

This analysis for *adder_9* on the Falcon and Sycamore machines is shown in Figure 6. The 1-qubit dependent region shrinks as the 1-qubit fidelity improves, even as we allow for any single-qubit gate count. At some 1-qubit fidelity, the region completely disappears. We denote this as the *1-qubit threshold fidelity*: once this is reached, no improvement in 1-qubit gate fidelity will improve the relative performance between two machines.

We can solve for the threshold fidelity for different initial 2-qubit gate counts, as present in algorithms. We set up

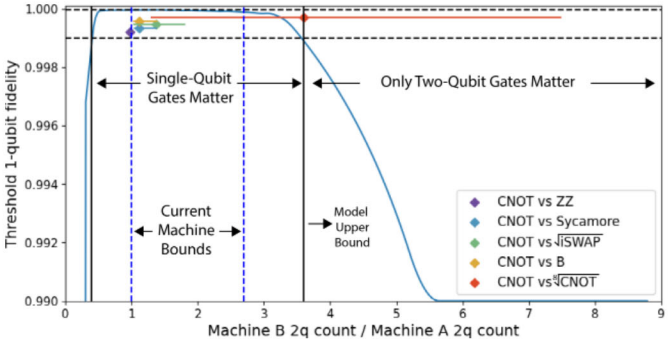

Fig. 7: *The X-axis shows the 2-qubit gate count ratio when comparing the implementations on two machines, while the Y-axis shows the resulting threshold 1-qubit fidelity. We also plot the range of 2-qubit gate count ratios we see for each gate compared to CNOT. The black dotted lines show the current NISQ single-qubit fidelity range. When using the upper bound on gate count ratio, the ordering of most machine comparisons is affected by the single-qubit gates. This quickly changes when we consider specific machines as shown in Table IV.*

the objective function such that the 2-qubit counts for each machine are related by a ratio, and we vary this ratio along the x-axis in Figure 7.

$$\pi = y^{n_1^A} \cdot f_2^{A^{n_2^A}} - y^{n_1^B} \cdot f_2^{B^{n_2^A \cdot x}}$$

As shown in Figure 7, there is an upper bound ratio (3.5) where the threshold fidelity drops below the current worst NISQ-era single-qubit fidelity! For any two circuits where the 2-qubit gate ratio is higher than the upper bound (3.5), no improvements in the 1-qubit gate count on any machine can change relative performance. This explains why for several circuit families, the CNOT machine always beats the $\sqrt[8]{\text{CNOT}}$ machine regardless of the single-qubit configuration!

We can then derive hardware-specific upper bound ratios which give direct information about the potential of changing relative performance in practice. Instead of using our generic bounds for our fidelity model, we use the 2-qubit gate fidelity ranges of existing machines and show results in Table IV. As

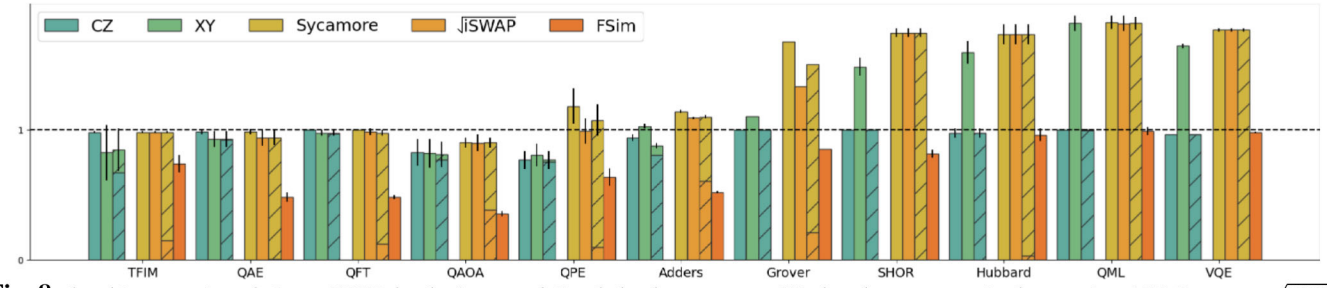

Fig. 8: *2-qubit gate ratios relative to CNOT, for the Aspen and Google hardware systems. We show homogeneous implementations (CZ, Sycamore, $\sqrt{iSWAP}$), heterogeneous implementations (CZ+XY, Sycamore+$\sqrt{iSWAP}$), and parameterized entangling gates (XY, FSim). Heterogeneous implementations (stacked bars) are as good as homogeneous ones. The parameterized FSim gate is able to express every algorithm well and is worth targeting.*

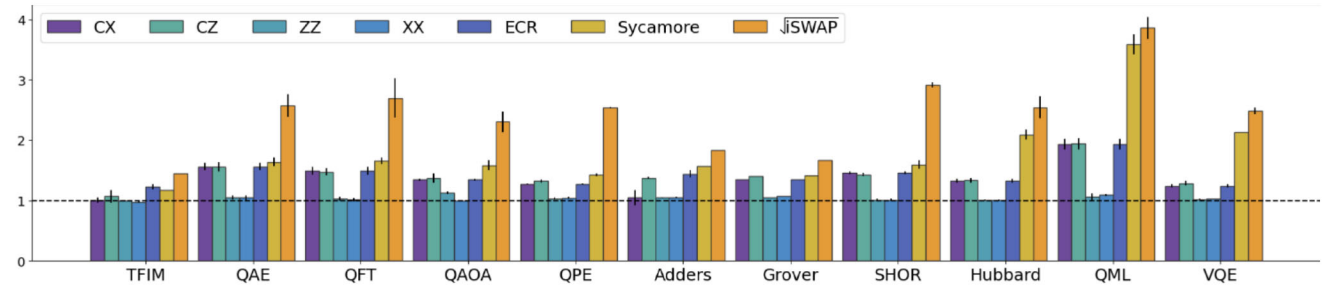

Fig. 9: *2-qubit gate counts averaged across circuit families mapped to each device topology and gate set.*

indicated, the actual upper bounds are around 1.6, much lower than the 3.5 absolute threshold. Overall, this data indicates that for most systems, relative performance orderings can be changed by tuning 1-qubit gates. However, when comparing QML or Shor circuits on Falcon and Sycamore machines, algorithm and compiler designers should not focus on 1-qubit gate reduction in order to improve relative performance.

D. Mixing Entangling Gates

Systems such as Aspen and Sycamore offer multiple entangling gates and parameterized entangling gates. These gates will have different average fidelities and may be used together within a single circuit. In Figure 8 we show the 2-qubit gate counts for the Sycamore and Aspen machines when using parameterized and multiple 2-qubit entangling gates within the same circuit. Heterogeneous gate sets like Sycamore+$\sqrt{\text{iSWAP}}$ and CZ+XY express circuits as well as the Sycamore or CZ gate alone, respectively. Therefore, on these systems choosing the highest fidelity gate for any algorithm may be sufficient.

The parameterized FSim gate leads to significant gate count reduction when compared to the Sycamore and $\sqrt{\text{iSWAP}}$ gates, while the XY Gate provides no benefit over the CZ gate. The FSim gate takes two parameters, while the XY gate only has one. This allows the FSim gate to express complex unitaries more efficiently. Accordingly, FSim implementations can outperform Sycamore gate implementations even with a larger (0.4%) drop in fidelity! The circuit quality of the FSim gate may also point to a finite *spanning gate set* of FSim family gates that are able to express circuits as well as the full parameterized gate. A finite set of constant gates may prove to be easier to calibrate than a fully parameterized gate, leading to a higher gate fidelity.

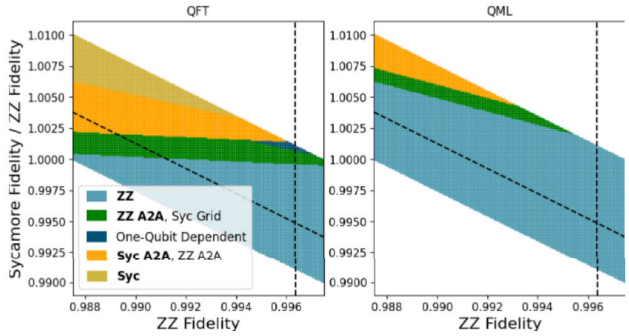

Fig. 10: *Roofline analysis (ZZ, Sycamore) is mapped to (all-to-all, mesh). Dark blue region: Relative performance can always be changed by tuning 1-qubit fidelity. Light Blue region: ZZ always performs best, regardless of topology. Gold region: Sycamore always performs best.*

E. Topology

The physical qubit interconnection topology impacts system behavior by:

1) Increasing gate counts when the algorithm logical topology is mismatching, as shown in Figure 9.
2) Adding cross talk due to the device qubit couplings.

Architects can use our models to answer: "What fidelity improvement is needed to overcome a more restrictive topology?". In Figure 10 we compare the capabilities of H2 (ZZ, all-to-all) with Sycamore (Sycamore, mesh) using the cyclic fidelity model. As before, we are able to identify ranges where one configuration (gate, topology) performs best irrespective of 1-qubit gate fidelity, as well as ranges where relative performance depends on the 1-qubit gate fidelity. The orange region in the graph shows the fidelity range in which Sycamore loses on account of being mapped to a more restrictive topology. The size of this orange region corresponds to the change in ability to express a circuit, and varies by algorithm.

While Figure 10 is able to model the first effect of topology, we must turn to the coupling-based model introduced in [15]

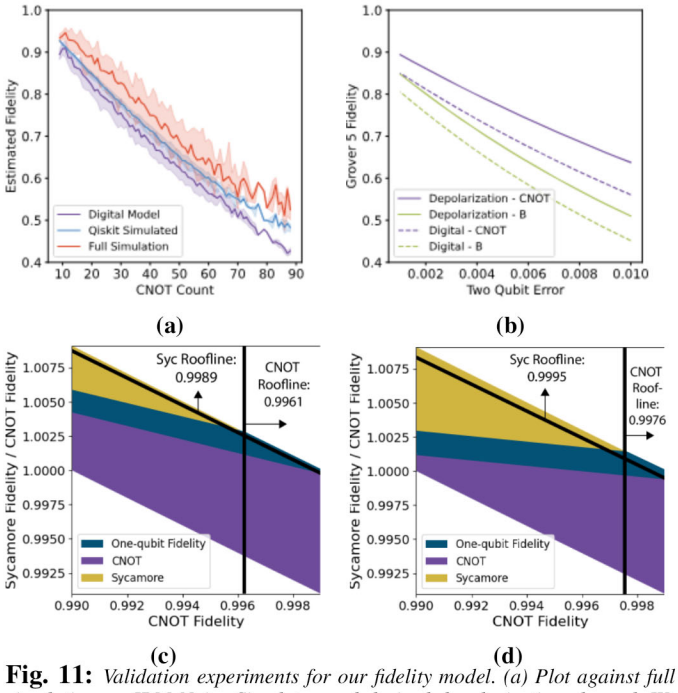

Fig. 11: *Validation experiments for our fidelity model. (a) Plot against full simulation on IBM Noisy Simulator and derived depolarization channel. We keep constant depth circuits with varied CNOT count gates. (b) Varying the 2-qubit gate error of our model vs. a general depolarization model. (c) and (d) 2-qubit gate analysis for adder_9 and grover_5 respectively using fully noisy simulation. We see that the roofline numbers and regions closely match the behavior seen in Figure 3.*

in order to understand cross talk:

$$\mathbf{F}_{\mathbf{c}_{\text{top}}} = (1 - e_{c1} \cdot C_1)^{n_1} \cdot (1 - e_{c2} \cdot C_2)^{n_2}$$

where C_i is the number of other qubits on average that each qubit is coupled to and e_{ci} is the error per coupling for an i-qubit gate. C_i is a direct measure of the physical chip topology. For a grid topology, usually associated with superconducting qubits, C_2 is a constant between 1 and 4. For all-to-all connectivities provided by ion traps C_2 is instead $\frac{1}{2}N(N-1)$. This error per coupling can be measured by the Cycle Benchmarking protocol.

VIII. Validation

These models trade off accuracy for tractability. While full noise simulation is able to give the most accurate view of algorithmic fidelity, it does not scale to system sizes of interest. However, we can use it as a point of comparison to validate the fidelity models we have chosen.

We first compare random circuits with varied gate count (see Figures 11a, 11b). We compare our digital model against two other models: full simulation and a depolarization channel model. We use Qiskit's provided noisy backends which account for T1 and T2 coherence times of qubits, as well as explicit error channels for each gate application [34]. We expect the fidelity model to closely lower bound both other procedures, which we see in Figure 11a. Secondly, we ensure that our model outputs similar relative results as we vary the 2-qubit error for different gate sets. We show these experiments in Figure 11b for the *grover_5* circuit transpiled to a CNOT machine and a B-gate machine. We see that the relative performance remains the same as we vary the error.

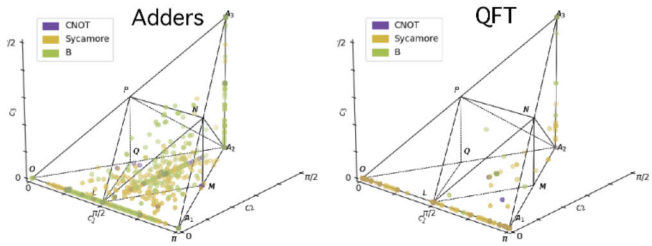

Fig. 12: *Position in the Weyl chamber of 2-qubit unitaries/blocks that arise in the Adder group (left) and QFT group (right) for different native gates. The initial spread of Adder unitaries proves to be a very difficult pattern for the Sycamore and B-gates to instantiate, while the periodic pattern we see in QFT (series of controlled rotations) are able to be expressed efficiently.*

We add further confirmation by running an full 2-qubit analysis as shown in Figure 3 using noisy simulation. As shown in Figure 11c, we see that the roofline numbers across both experiments closely match. We feel that the correlations seen in our model across these experiments in addition to the correlation seen in experiments run on real hardware [1], [15] validate the utility of our fidelity model.

IX. Evaluating Gate Representational Power

A gate set's ability to realize a circuit comes down to its expressivity and entanglement. Gate *expressivity* identifies a gate's ability to represent a random two-qubit unitary. Architects often use the Weyl Chamber to directly visualize gate expressivity [24]. The Weyl Chamber removes all local parameters from a 2-qubit unitary and plots it into a tetrahedron. Most entangling gates can express any 2-qubit unitary in three applications (along with single qubit rotation gates). The most expressive gate, the B-gate, can express any unitary in two applications. It can also easily be realized on a superconducting machine [46]. Gate *entanglement* is a gate's ability to maximize the entanglement between two qubits. CNOTs and most hardware native gates are maximally entangling: a single application to two qubits will leave them perfectly correlated. The $\sqrt[4]{\text{CNOT}}$ and $\sqrt[8]{\text{CNOT}}$ gates trade off entangling power for large potential gains in gate fidelity.

Several trends have become apparent in our research. Most notably, we see that current state-of-the-art compilers are unable to generate higher quality circuits when targeting the B-gate. This is surprising, because the B-gate is the most expressive 2-qubit gate. On the other hand, we see some equally surprising positive results for the $\sqrt[4]{\text{CNOT}}$ and $\sqrt[8]{\text{CNOT}}$ gates. These low entangling gates require 4 and 8 applications respectively to represent a single CNOT gate. This makes their comparable expressivity in important circuits such as QFT, TFIM, and QAE circuits exciting.

A. Weyl Chamber Distribution of Two-Qubit Algorithm Blocks

Under our procedure, it is clear that the expressivity and entanglement of a native 2-qubit gate is not strongly correlated with algorithm performance under currently accepted design criteria. While expressivity is assessed based on the power to implement random 2-qubit unitaries, optimal implementations

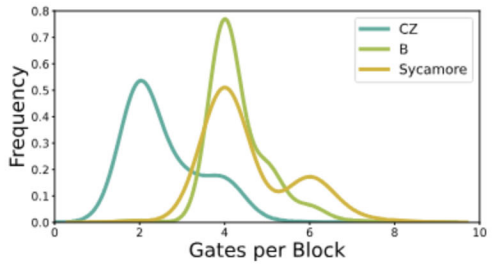

Fig. 13: *The BQSKit partitioner splits the circuit into circuit blocks. We plot the number of 2-qubit gates (CZ, B, Sycamore) in each block of the Hubbard circuits as a distribution. Note that the majority of blocks are very simple (count is < 4 for CZ). These blocks are too simple for the B gate to find smaller circuits, and usually defaults to using 2 B gates to represent a single CZ. On the other hand, the B gate is able to find shorter circuits for the rare longer blocks in the circuit.*

of algorithms impose structure on the set of 2-qubit unitaries that these gates decompose.

In order to analyze the distribution of 2-qubit unitaries that appear in different algorithm classes, we plot the position of these unitaries on the Weyl chamber (Figure 12). We compare the position of points after compiling to different gate sets. As seen in the figure, the input structure of the Adder group, while not random, is diverse. Both the B gate and Sycamore gate struggle to represent this distribution and their resultant distribution is even more spread out. This occurs as the new gate sets are unable to map the original distribution of unitaries, and generate new two-qubit unitaries that are less efficient in terms of gate count. On the other hand, the QFT has incredibly periodic unitaries. These unitaries are much simpler to translate to any gate set, and we see that the resultant spread is diminished.

B. Synthesis Derived Gate Selection Criteria

The incorporation of circuit synthesis in our compilation workflow enables us to derive additional criteria for gate selection and development. The advantages derived from our flow are due to synthesis' powerful compilation capabilities.

Given an input circuit, traditional compilers will use local peepholes optimization, translating from one gate set to another using analytical, one-to-one gate rewriting rules for 2-qubit gates. For example, a CNOT is translated into a sequence of two Sycamore gates and additional U3 gates. One-to-one gate rewriting has been shown to be less than optimal [44]. BQSkit's synthesis based compiler [45] employs a different strategy. Given an input circuit, BQSKit partitions it into multi-qubit blocks (partitions). Each partition is optimized and translated using optimal topology aware direct unitary synthesis [13], combined with a powerful synthesis based mapping and routing algorithm [25]. Thus, deploying synthesis leads to different conclusions than when using vendor provided compilers. We see that the Sycamore roofline fidelity completely disappear when running the analysis shown in Figure 3, while the CNOT roofline drops to 0.995.

More insights can be gained by examining gate representational power to implement 3-qubit blocks/processes that arise in algorithm implementations. We use the BQSKit partitioner to decompose a circuit into maximal 3-qubit gates and then use direct synthesis to generate circuits targeting each native gate set. This procedure results in implementations for each block that use the fewest number of 2-qubit gates, irrespective of gate choice. We plot the gate count distribution of blocks in Figure 13. The CNOT family of gates (which includes the CZ gate) unsurprisingly have a one-to-one mapping, but the story changes for the Sycamore and B-gates. Using Sycamore gates increases gate count, which is to be expected. On the other hand, we see that the B-gate is able to better express some more complicated blocks (5-7 CNOT blocks reduce to 4 B-gates). However, the overall gate count reduction is held back by the B-gate's inability to express simpler blocks efficiently. While a larger block granularity (4+ qubits) would not remove all simple blocks from these circuits, it remains to be seen whether the average increased complexity in each block would allow the B-gate to outperform other gates.

Overall, this analysis indicates that existing gate design criteria should be augmented. In addition to choosing a gate based on attainable fidelity and its representational power for *random* 2-qubit unitaries, the gate representational power for multi-qubit blocks (e.g. three qubit unitaries) drawn from implementations of real workloads should be considered.

X. Conclusion

While we have introduced and examined several fidelity models, we have emphasized the derivation of the roofline approach for the digital model. A similar approach can be taken for the cyclic model. Each cycle's fidelity has an absolute parallelism threshold of $\frac{1}{P_i}$ according to the model, and this number will reduce as specific machines/circuits are targeted.

We have also targeted our roofline model for hardware improvement and algorithmic development by considering relative gate fidelity across two configurations. By changing emphasis from fidelity to gate count and circuit depth, a similar derivation can produce roofline models for compiler developers to guide circuit optimization decisions: "What mix of gates to choose?; "Should I reduce gate count or increase gate parallelism?" etc.

This paper extends the idea of quantum HW/SW co-design [23], [38] through the consideration of device gate sets that target specific algorithms. For TFIM, QFT, and QAE circuits, we have shown that a designer should maximize gate fidelity even at the cost of expressivity and entanglement capability. On the other hand, we see that highly expressive gates such as the B-gate provide little improvement in overall circuit fidelity. Restricting the topology from an all-to-all connectivity leads to a potentially massive need for higher gate fidelity, depending on the algorithm. This means that for Adder-like circuits, Hubbard models, or QML networks, an ion trap machine with a ZZ or XX gate is best suited. Our results indicate that unlike classical benchmarking which is compiler independent, quantum system evaluation and benchmarking is sensitive to the quality of compilation tools. For the time being, compilation workflows require circuit synthesis in order to make robust inferences.

We believe our methodology will apply beyond the NISQ era into the Fault Tolerant (FT) quantum regime. While for NISQ we directly minimize two-qubit gate count, compilers targeting FT applications must optimize the number of non-transversal gates (typically T, Toffoli, or CCZ gates) as well as the area, latency and error correction overhead [30]. It has been shown that algorithm-dependent design significantly impacts resource utilization [6], and this lends itself perfectly for our roofline analysis.

In summary, we introduce a procedure for performing comparisons between quantum system configurations. In our quantum roofline analysis, we derive bounds on system properties (e.g. gate fidelity) that can be used as a stop criteria for optimization efforts. We then evaluate machines across a large set of important algorithms and are able to quantify the trade-off required between gate fidelity, expressivity, and entanglement for different circuit families in order to maximize circuit execution fidelity. Our work also shows that the ability of circuit synthesis to generate resource minimal circuits is paramount to performance evaluation, and it enables new design criteria for gate set adoption. We believe our procedure is of interest not only to hardware designers, but compiler and algorithm developers as well.

REFERENCES

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. R. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. N. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandra, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, "Supplementary information for "Quantum supremacy using a programmable superconducting processor"," *Nature*, vol. 574, no. 7779, pp. 505–510, Oct. 2019, arXiv:1910.11333 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1910.11333

[2] *Highly Scalable Quantum Computing Wth Atomic Arrays*, Atom Computing, 2023. [Online]. Available: https://atom-computing.com/wp-content/uploads/2022/08/Atom-Computing-Atomic-Arrays.pdf

[3] F. Bao, H. Deng, D. Ding, R. Gao, X. Gao, C. Huang, X. Jiang, H.-S. Ku, Z. Li, X. Ma, X. Ni, J. Qin, Z. Song, H. Sun, C. Tang, T. Wang, F. Wu, T. Xia, W. Yu, F. Zhang, G. Zhang, X. Zhang, J. Zhou, X. Zhu, Y. Shi, J. Chen, H.-H. Zhao, and C. Deng, "Fluxonium: an alternative qubit platform for high-fidelity operations," *Physical Review Letters*, vol. 129, no. 1, p. 010502, Jun. 2022, arXiv:2111.13504 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2111.13504

[4] R. Barends, J. Kelly, A. Megrant, D. Sank, E. Jeffrey, Y. Chen, Y. Yin, B. Chiaro, J. Mutus, C. Neill, P. O'Malley, P. Roushan, J. Wenner, T. C. White, A. N. Cleland, and J. M. Martinis, "Coherent josephson qubit suitable for scalable quantum integrated circuits," *Phys. Rev. Lett.*, vol. 111, p. 080502, Aug 2013. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.111.080502

[5] S. Beauregard, "Circuit for Shor's algorithm using 2n+3 qubits," Feb. 2003, arXiv:quant-ph/0205095. [Online]. Available: http://arxiv.org/abs/quant-ph/0205095

[6] M. E. Beverland, P. Murali, M. Troyer, K. M. Svore, T. Hoefler, V. Kliuchnikov, G. H. Low, M. Soeken, A. Sundaram, and A. Vaschillo, "Assessing requirements to scale to practical quantum advantage," 2022.

[7] S. Bravyi and A. Kitaev, "Fermionic quantum computation," *Annals of Physics*, vol. 298, no. 1, pp. 210–226, May 2002, arXiv:quant-ph/0003137. [Online]. Available: http://arxiv.org/abs/quant-ph/0003137

[8] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles, "Challenges and opportunities in quantum machine learning," *Nature Computational Science*, vol. 2, no. 9, pp. 567–576, Sep. 2022, number: 9 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s43588-022-00311-3

[9] J.-S. Chen, E. Nielsen, M. Ebert, V. Inlek, K. Wright, V. Chaplin, A. Maksymov, E. Páez, A. Poudel, P. Maunz, and J. Gamble, "Benchmarking a trapped-ion quantum computer with 29 algorithmic qubits," Aug. 2023, arXiv:2308.05071 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2308.05071

[10] J. I. Cirac and P. Zoller, "Quantum computations with cold trapped ions," *Phys. Rev. Lett.*, vol. 74, pp. 4091–4094, May 1995. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.74.4091

[11] A. Cowtan, S. Dilkes, R. Duncan, W. Simmons, and S. Sivarajah, "Phase gadget synthesis for shallow circuits," *Electronic Proceedings in Theoretical Computer Science*, vol. 318, p. 213–228, May 2020. [Online]. Available: http://dx.doi.org/10.4204/EPTCS.318.13

[12] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, "Validating quantum computers using randomized model circuits," *Physical Review A*, vol. 100, no. 3, p. 032328, Sep. 2019, arXiv:1811.12926 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1811.12926

[13] M. G. Davis, E. Smith, A. Tudor, K. Sen, I. Siddiqi, and C. Iancu, "Towards optimal topology aware quantum circuit synthesis," in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2020, pp. 223–234.

[14] C. Developers, "Cirq," Jul. 2023. [Online]. Available: https://doi.org/10.5281/zenodo.8161252

[15] A. Erhard, J. J. Wallman, L. Postler, M. Meth, R. Stricker, E. A. Martinez, P. Schindler, T. Monz, J. Emerson, and R. Blatt, "Characterizing large-scale quantum computers via cycle benchmarking," *Nature Communications*, vol. 10, no. 1, nov 2019. [Online]. Available: https://doi.org/10.1038%2Fs41467-019-13068-7

[16] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm," Nov. 2014, arXiv:1411.4028 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1411.4028

[17] D. Herman, C. Googin, X. Liu, A. Galda, I. Safro, Y. Sun, M. Pistoia, and Y. Alexeev, "A Survey of Quantum Computing for Finance," Jun. 2022, arXiv:2201.02773 [quant-ph, q-fin]. [Online]. Available: http://arxiv.org/abs/2201.02773

[18] P. Horodecki, M. Horodecki, and R. Horodecki, "General teleportation channel, singlet fraction and quasi-distillation," Mar. 1999, arXiv:quant-ph/9807091. [Online]. Available: http://arxiv.org/abs/quant-ph/9807091

[19] IonQ Staff, "IonQ Forte: The First Software-Configurable Quantum Computer." [Online]. Available: https://ionq.com/resources/ionq-forte-first-configurable-quantum-computer

[20] D. Jaksch, J. I. Cirac, P. Zoller, S. L. Rolston, R. Côté, and M. D. Lukin, "Fast quantum gates for neutral atoms," *Phys. Rev. Lett.*, vol. 85, pp. 2208–2211, Sep 2000. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.85.2208

[21] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland, "Randomized Benchmarking of Quantum Gates," *Physical Review A*, vol. 77, no. 1, p. 012307, Jan. 2008, arXiv:0707.0963 [quant-ph]. [Online]. Available: http://arxiv.org/abs/0707.0963

[22] G. Li, Y. Ding, and Y. Xie, "Tackling the qubit mapping problem for nisq-era quantum devices," 2019.

[23] G. Li, A. Wu, Y. Shi, A. Javadi-Abhari, Y. Ding, and Y. Xie, "On the Co-Design of Quantum Software and Hardware," in *Proceedings of the Eight Annual ACM International Conference on Nanoscale Computing and Communication*, ser. NANOCOM '21. New York, NY, USA: Association for Computing Machinery, Sep. 2021, pp. 1–7. [Online]. Available: https://dl.acm.org/doi/10.1145/3477206.3477464

[24] S. F. Lin, S. Sussman, C. Duckering, P. S. Mundada, J. M. Baker, R. S. Kumar, A. A. Houck, and F. T. Chong, "Let each quantum bit choose its basis gates," in *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2022, pp. 1042–1058.

[25] J. Liu, E. Younis, M. Weiden, P. Hovland, J. Kubiatowicz, and C. Iancu, "Tackling the Qubit Mapping Problem with Permutation-Aware Synthesis," May 2023, arXiv:2305.02939 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2305.02939

[26] E. Magesan, J. M. Gambetta, and J. Emerson, "Robust randomized benchmarking of quantum processes," *Physical Review Letters*, vol. 106, no. 18, p. 180504, May 2011, arXiv:1009.3639 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1009.3639

[27] E. Magesan, J. M. Gambetta, and J. Emerson, "Characterizing Quantum Gates via Randomized Benchmarking," *Physical Review A*, vol. 85, no. 4, p. 042311, Apr. 2012, arXiv:1109.6887 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1109.6887

[28] A. Mandviwalla, K. Ohshiro, and B. Ji, "Implementing Grover's Algorithm on the IBM Quantum Computers," in *2018 IEEE International Conference on Big Data (Big Data)*, Dec. 2018, pp. 2531–2537. [Online]. Available: https://ieeexplore.ieee.org/document/8622457

[29] J. R. McClean, I. D. Kivlichan, D. S. Steiger, Y. Cao, E. S. Fried, C. Gidney, T. Häner, V. Havlíček, Z. Jiang, M. Neeley, J. Romero, N. Rubin, N. P. D. Sawaya, K. Setia, S. Sim, W. Sun, K. Sung, and R. Babbush, "Openfermion: The electronic structure package for quantum computers," 2017, cite arxiv:1710.07629. [Online]. Available: http://arxiv.org/abs/1710.07629

[30] M. Möttönen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, "Quantum circuits for general multiqubit gates," *Physical review letters*, vol. 93, no. 13, p. 130502, 2004.

[31] M. A. Nielsen, "A simple formula for the average gate fidelity of a quantum dynamical operation," *Physics Letters A*, vol. 303, no. 4, pp. 249–252, oct 2002. [Online]. Available: https://doi.org/10.1016%2Fs0375-9601%2802%2901272-0

[32] L. B. Oftelie, R. V. Beeumen, E. Younis, E. Smith, C. Iancu, and W. A. de Jong, "Constant-depth circuits for dynamic simulations of materials on quantum computers," *Materials Theory*, vol. 6, no. 1, mar 2022. [Online]. Available: https://doi.org/10.1186%2Fs41313-022-00043-x

[33] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, "A variational eigenvalue solver on a quantum processor," *Nature Communications*, vol. 5, no. 1, p. 4213, Jul. 2014, arXiv:1304.3061 [physics, physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/1304.3061

[34] Qiskit contributors, "Qiskit: An open-source framework for quantum computing," 2023.

[35] *Quantinuum System Model H2*, Quantinuum, 2023.

[36] P. Rakyta and Z. Zimborás, "Efficient quantum gate decomposition via adaptive circuit compression," 2022.

[37] *Aspen-M-3 Quantum Processor*, Rigetti QCS, 2023.

[38] H. Safi, K. Wintersperger, and W. Mauerer, "Influence of HW-SW-Co-Design on Quantum Computing Scalability," Jun. 2023, arXiv:2306.04246 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2306.04246

[39] D. Shin, H. Hübener, U. De Giovannini, H. Jin, A. Rubio, and N. Park, "Phonon-driven spin-Floquet magneto-valleytronics in MoS2," *Nature Communications*, vol. 9, no. 1, p. 638, Feb. 2018, number: 1 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41467-018-02918-5

[40] S. Sivarajah, S. Dilkes, A. Cowtan, W. Simmons, A. Edgington, and R. Duncan, "tket: A Retargetable Compiler for NISQ Devices," *Quantum Science and Technology*, vol. 6, no. 1, p. 014003, Jan. 2021, arXiv:2003.10611 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2003.10611

[41] R. R. Tucci, "An introduction to cartan's kak decomposition for qc programmers," 2005.

[42] M. Weiden, E. Younis, J. Kalloor, J. Kubiatowicz, and C. Iancu, "Improving Quantum Circuit Synthesis with Machine Learning," Jun. 2023, arXiv:2306.05622 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2306.05622

[43] S. Williams, A. Waterman, and D. Patterson, "Roofline: an insightful visual performance model for multicore architectures," *Commun. ACM*, vol. 52, no. 4, p. 65–76, apr 2009. [Online]. Available: https://doi.org/10.1145/1498765.1498785

[44] E. Younis and C. Iancu, "Quantum Circuit Optimization and Transpilation via Parameterized Circuit Instantiation," Jun. 2022, arXiv:2206.07885 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2206.07885

[45] E. Younis, C. C. Iancu, W. Lavrijsen, M. Davis, E. Smith, and USDOE, "Berkeley quantum synthesis toolkit (bqskit) v1," 4 2021. [Online]. Available: https://www.osti.gov//servlets/purl/1785933

[46] J. Zhang, J. Vala, S. Sastry, and K. B. Whaley, "Minimum construction of two-qubit quantum operations," *Physical Review Letters*, vol. 93, no. 2, p. 020502, Jul. 2004, arXiv:quant-ph/0312193. [Online]. Available: http://arxiv.org/abs/quant-ph/0312193