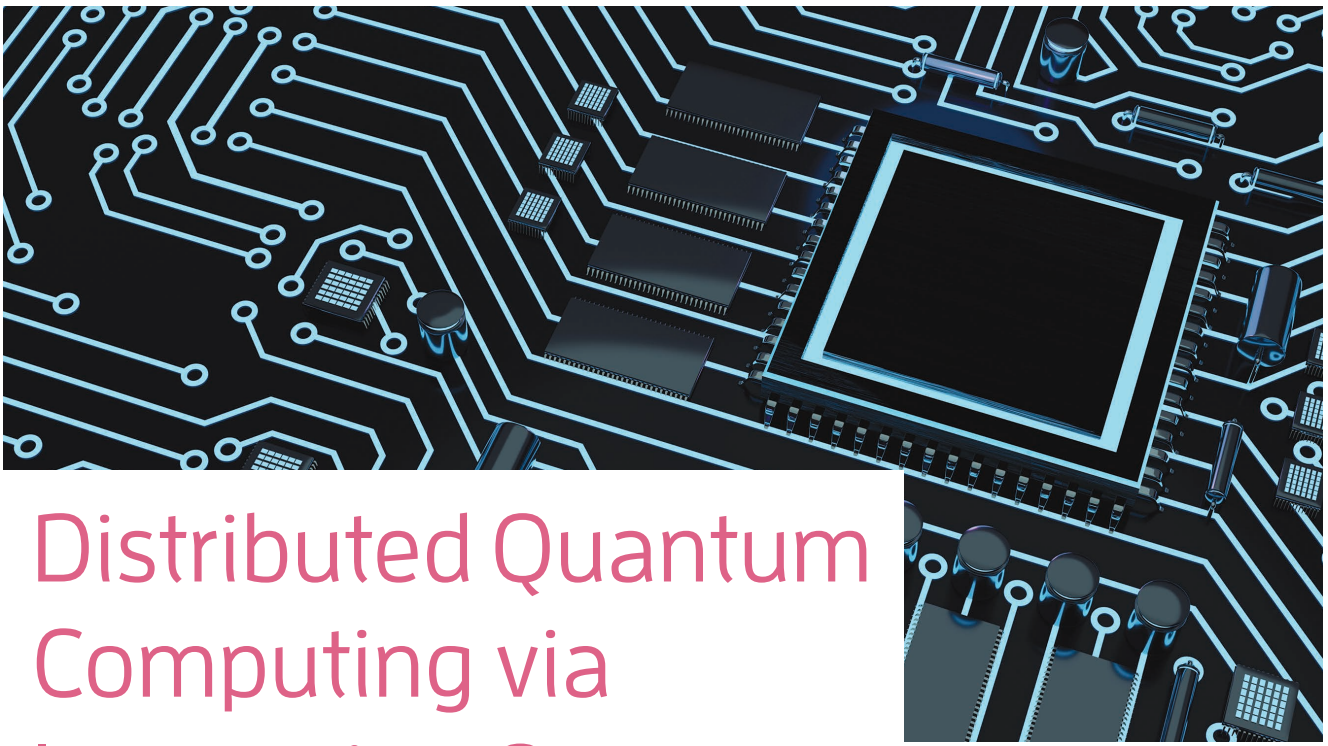
EDITORS
TIMOTHY JONES University of Cambridge, U.K., timothy.jones@cl.cam.ac.uk
ROBERT MULLINS University of Cambridge, U.K., robert.mullins@cl.cam.ac.uk



Distributed Quantum Computing via Integrating Quantum and Classical Computing

Wei Tang and **Margaret Martonosi**, Princeton University

As quantum computing confronts scalability challenges, distributed hybrid QPU–CPU techniques emerge as a crucial solution. These techniques distribute quantum algorithms across quantum and classical computing resources to surpass the computational reach of either one alone.

Digital Object Identifier 10.1109/MC.2024.3360569
Date of current version: 5 April 2024

As we navigate an era where the insatiable demand for computing power is increasingly outpacing the capabilities of traditional computing, the slowing progress of Moore's law poses a formidable challenge. At this critical juncture, quantum computing (QC) emerges with the potential to greatly extend computing capabilities in key application domains.

Unlike classical computing operating on binary information, QC rests on quantum bits, or *qubits*, exploiting the nonintuitive yet powerful properties of quantum mechanics, such as entanglement and superposition. This gives QC the potential to perform complex calculations at speeds unattainable by its classical counterparts, toward solving problems considered intractable today. However, scaling up QC systems to these levels involves overcoming substantial engineering and scientific

challenges. Current efforts in QC primarily concentrate on enhancing the performance of a single quantum processing unit (QPU). The goal of these efforts is to increase both QPU size and precision, paving the way for QC to tackle real-world applications that were once thought beyond reach.

By contrast, our work seeks to exploit distributed computing that hybridizes quantum and classical approaches. In today's commercial world, distributed and parallel classical computing is not just conceptual; it's an integral part of our daily tech interactions. Distributed classical computing divides complex tasks across multiple computers for simultaneous processing. Unlike the case of a single supercomputer handling all of the tasks, distributed computing spreads these tasks across several, or sometimes thousands of, CPUs, GPUs, and other units. Video games, high-resolution video editing, and artificial intelligence are just a few examples that leverage the collective power of numerous CPUs and GPUs to tackle tasks that would be impossible for a single machine.

The aim of this article is to provide an in-depth exploration of the emerging field of *distributed QC*, particularly through the lens of integrating quantum and classical computing paradigms. In classical distributed computing, data parallelism and model parallelism are two key strategies for processing large or complex tasks. Data parallelism divides a large dataset into smaller chunks, distributing them across multiple nodes for parallel processing. Each node works on its data segment independently, necessitating the initial distribution of these data parts and possibly aggregating the results later. Another approach, model parallelism, involves splitting a complex model, such as a neural network, across nodes, with each working on a different part. This strategy requires a continuous exchange of intermediate results among nodes for collaborative processing.

At first glance, distributing QC tasks across multiple nodes seems a straightforward extension of classical distributed computing. However, the field faces a unique challenge rooted in the fundamental laws of quantum physics. The quantum no-cloning theorem,[5] a cornerstone principle in quantum mechanics, dictates that it is impossible to create an exact copy of an arbitrary unknown quantum state. This prohibition against duplicating quantum data presents a significant obstacle in developing distributed QC systems as it contradicts the typical data-sharing methodologies employed in classical distributed computing. This article plots a way forward for navigating this challenge, exploring innovative approaches to distribute quantum tasks without copying quantum data. We examine the intersection of quantum and classical computing techniques, shedding light on how this hybrid model can potentially unlock new capabilities and applications in the QC landscape.

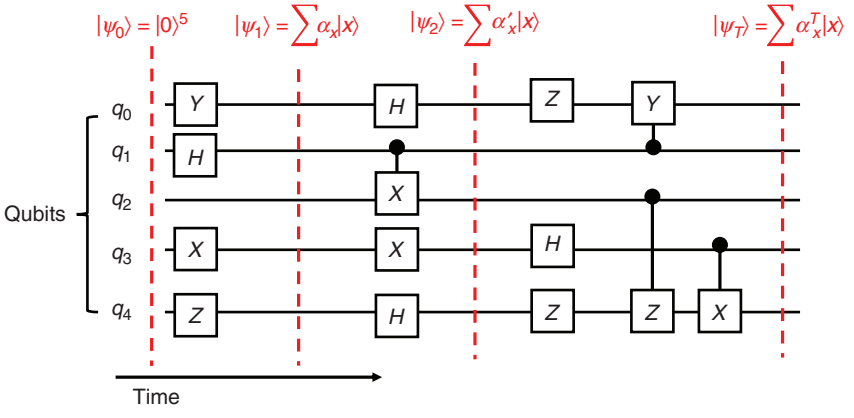


FIGURE 1. Example of a five-qubit quantum circuit. Final measurements sample and compute the α_x^T coefficients, or amplitudes, for each possible quantum state, representing a target solution.

BACKGROUND

Unlike classical computers, which use binary 0 or 1 bits as the smallest unit of data, quantum computers use quantum bits, or "qubits." Qubits have the unique ability to exist in multiple states at once, thanks to a quantum phenomenon known as superposition. Imagine a coin that can spin in the air without ever landing—its state is a probabilistic superposition of the heads and tails outcomes it might land as. Another principle is entanglement, where two qubits become linked in such a way that the state of one can instantly affect the state of another, regardless of the distance between them. Knowing whether one coin lands as heads or tails offers the observer information about the state of another entangled coin, even if distant. These properties are not observable at the scale of objects like coins, but they are real physical properties that are measurable at the atomic scale. By exploiting these properties, quantum computers can perform complex calculations at incredible scale, potentially solving problems that are currently intractable for classical computers.

At its core, a quantum program is expressed as a circuit composed of a sequence of quantum operations, known as *gates*, which act on the qubits. These gates, which can be single-qubit or multiqubit operations, play a crucial role in manipulating the states of the qubits. Figure 1 shows an example quantum circuit with five qubits. Each horizontal line denotes a qubit. Boxes incident on a single qubit wire are single-qubit quantum gates, which operate on that qubit. Boxes incident on two qubit wires are two-qubit quantum gates, which operate on both of them.

As the circuit progresses, gates are applied in sequence, altering the collective quantum states of the qubits. This process starts from an initial quantum state on the left-hand side and evolves toward the desired final quantum state on the right. To control this process, sophisticated control electronics are used. These electronics manage the timing, order, and type of quantum gates applied to the qubits. They often include precision timing equipment and can include microwave pulse generators for systems like superconducting qubits, or laser systems for ion-trap qubits.

The process terminates with measuring the qubits, which produces a classical binary string—a phenomenon known as the collapse of the quantum state. Because quantum states and operations are probabilistic in nature, the quantum circuit is executed multiple times. This repetition allows for the accumulation of statistical data, reflecting the final amplitude coefficients, denoted as α_x^T, for all possible quantum states. It is through this meticulous and repeated application of quantum gates and measurements that the quantum circuit unveils the solution encoded in the target quantum state. The illustrated circuit requires a QPU with at least five "good enough qubits" and "accurate-enough operations" to execute all of the quantum gates before too many errors accumulate to produce useful results.

Toward distributed QC

Traditionally, QC has primarily concentrated on the development and optimization of a single QPU, with the aspiration that it will eventually become sufficiently large and accurate to execute complex quantum circuits of practical significance. However, this approach encounters a formidable scalability challenge: many practical quantum applications require thousands of high-quality qubits. These are either implemented logically as error-corrected versions of millions of noisy and error-prone physical qubits, *or* the underlying physical qubits must have sufficient fidelity to avoid the need for error correction. Either way, the scale and complexity introduces significant engineering obstacles, making the realization of practical QC applications a daunting task. Against this backdrop, the role of distributed QC—which harnesses the collective power of multiple QPUs to share and process quantum workloads—has become increasingly crucial. By adopting a distributed framework, the scaling challenges of a single-QPU system can be substantially mitigated, greatly enhancing the potential scope and impact of QC.

TOWARD DISTRIBUTED QC: WHAT IS CIRCUIT CUTTING?

Achieving the goals of QC requires a practical solution to its challenges by breaking down large, complex quantum circuits into smaller, more manageable subcircuits. Circuit cutting is an innovative technique[2] that offers practical approaches for the individual subcircuits to be processed on different QPUs in parallel, and for the classical reconstruction phase that has traditionally stymied QC circuit-cutting techniques.

At the heart of circuit cutting is the concept of decomposition: it essentially involves identifying specific points, known as cut points, within a quantum circuit and then decomposing the complex quantum states at these points into a series of classical components based on a mathematical framework known as *Pauli bases*. Once each subcircuit is run on a QPU, the original, full quantum state must be reconstructed through classical post-processing, where the results of the separate subcircuits are combined in a specific and compute-intensive way. Naive implementations of classical reconstruction involve matrix multiplications and scale exponentially with factors like qubit state and cut points. Our work improves on these naive reconstruction approaches. By enabling QC circuit decomposition and by helping the subsequent reassembly of quantum tasks to be more tractable, circuit cutting effectively bridges the gap between the current capabilities of quantum hardware and the demands of complex quantum computations, making it a key technique in advancing the field of QC.

A QC circuit-cutting example

Figure 2 illustrates the process of circuit cutting using the straightforward quantum circuit example from Figure 1. In this example, circuit cutting is applied by making a strategic cut, denoted by a red cross, effectively dividing the original circuit into two smaller subcircuits.

The real power of circuit cutting is showcased in the next step, where these subcircuits are assigned to multiple three-qubit QPUs. These three-qubit QPUs only need to support the smaller subcircuits, hence placing fewer requirements on hardware quality. In addition, this approach introduces flexibility and efficiency as these QPUs can operate the subcircuits independently and in parallel, without the need for direct quantum communication between them. This independence is possible because the subcircuits are entirely decoupled by the cut.

What does a cut actually mean? At the red X, circuit cutting requires us to mathematically decompose the quantum state at the cut point into its four Pauli bases in $\{I, X, Y, Z\}$. This mathematical decomposition then allows classical computing to reconstruct the quantum state after QPU execution. Circuit cutting is thus characterized by making vertical cuts across the qubit wires, effectively segmenting a large quantum circuit into several smaller parts. In more complex scenarios, a large circuit might be divided using multiple cuts, further breaking down the computational task into even smaller subcircuits. This technique not only makes quantum computations more feasible on current quantum hardware but also significantly expands the range of problems that can be tackled using available QC resources.

Classical reconstruction postprocessing

As previously noted, straightforward or naive classical reconstruction methods are computationally expensive, bordering on intractable. For example, early proposals for circuit cutting,[2,4] while straightforward in approach and feasible to implement, involve a series of computationally intensive steps. Consider the method of Tang et al.,[4] which at its core requires the computation of the tensor product of the outputs from each subcircuit corresponding to a specific Pauli basis. This process is not a one-off calculation but must be repeated *for every possible combination of Pauli bases*. Once these tensor products for each Pauli basis combination are calculated, the next step involves summing up all of these intermediate results to achieve the final output. The complexity of this method becomes evident when considering the number of Pauli bases involved: with four Pauli bases associated with each cut, the total number of tensor product calculations needed grows exponentially with the addition of each cut.

Challenges

Circuit cutting hence encounters two significant challenges. The first challenge lies in the scalability of the approach. The initial theoretical proposal[2] for circuit cutting included a reconstruction formula for reassembling the final quantum state from the subcircuits that scales exponentially with the number of cuts made in the circuit. This exponential scaling poses a significant obstacle to the practical implementation of circuit cutting, especially for very large and complex quantum circuits.

Second, identifying optimal cut points within large quantum circuits is a complex task. The process involves not just splitting the circuit, but doing so in a manner that ensures each resulting subcircuit is computationally manageable and capable of being processed independently. This requires a careful balance between the complexity of individual subcircuits and the overall efficiency of computation.

The current state of circuit cutting

The first comprehensive implementation of circuit cutting[4] marks a significant advancement in this field by introducing an automated solver algorithm. This algorithm is designed to determine the minimal number of cuts necessary to divide a large quantum circuit into smaller subcircuits, which can then be processed on available smaller scale QPUs. To achieve this, the problem of finding these optimal cuts is formulated as a mixed-integer programming problem, enabling a more systematic and efficient approach to circuit segmentation.

A key aspect of this implementation is its adoption of a relatively straightforward method for the classical reconstruction of the quantum state postcomputation. However, minimizing the number of cuts becomes a critical objective. The reason is that beyond a certain circuit complexity level in terms of size or connectivity, the time and resources required for classical postprocessing overshadow the benefits gained from dividing the circuit, turning it into a computational bottleneck. The success of this implementation, therefore, hinges on striking a delicate balance—optimizing the circuit to fit smaller QPUs through the fewest possible cuts while keeping the postprocessing demands manageable to make circuit cutting a viable and practical approach in QC. Specifically, the work of Tang et al.[4] demonstrates running circuits up to 100 qubits.

Circuit cutting with tensor contraction

The state-of-the-art circuit-cutting technique[3] proposes to integrate distributed QC with classical tensor networks to exponentially improve the postprocessing process, hence eliminating the major obstacle for practical circuit cutting. Tensor networks have been widely used in classical simulations of quantum systems.[1] At its core, a tensor network consists of tensors (multidimensional arrays of numbers) connected by edges,

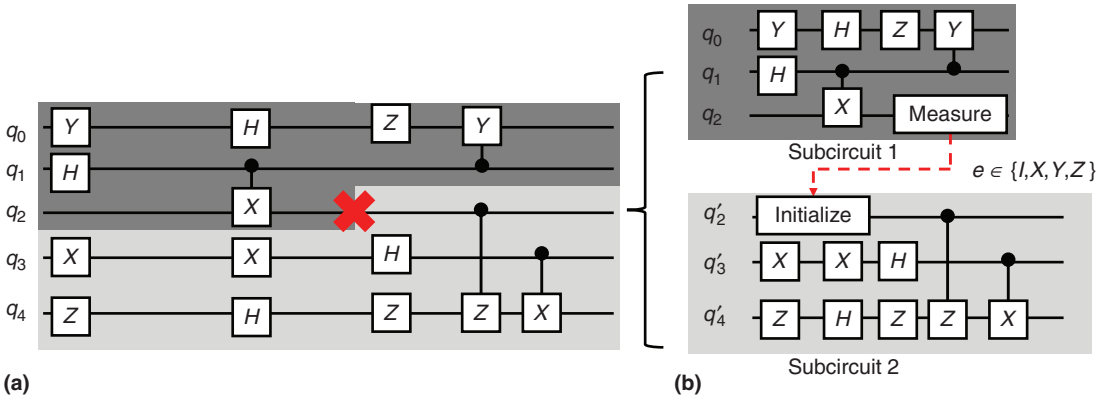


FIGURE 2. Example of cutting a five-qubit quantum circuit with one cut that divides it into two smaller subcircuits. (a) The red cross indicates the cutting point. Subcircuit 1 is shaded dark, and subcircuit 2 is shaded light. (b) Subcircuit 1 with measurements and subcircuit 2 with initialization in each one of the Pauli bases. The two subcircuits require no quantum communications and are executed independently in any order on multiple three-qubit QPUs.

where each edge represents a shared dimension or index between the tensors.

Tensor network contraction is a computational process in which the tensors in a network are systematically combined, or "contracted," according to specific rules. This contraction involves summing over shared indices or dimensions between connected tensors, effectively reducing the network into a single tensor to represent the results of the original network. On the other hand, classical reconstruction for circuit cutting also involves multiplying the subcircuit results across their shared cut qubit wires and taking the summation. Figure 3 shows the mathematical equivalence between the two processes.

Utilizing tensor network contraction in circuit cutting offers an exponential computational advantage over the simple brute-force reconstruction method, primarily because of its efficiency in managing high-dimensional data. In brute-force reconstruction, the process involves calculating and summing the tensor products for each possible combination of Pauli bases across all cuts, leading to an exponential increase in computations with the addition of each cut. By contrast, tensor networks contract the subcircuit tensors along their shared dimensions; this method effectively consolidates the network, step by step, into a single tensor.

This approach dramatically reduces the number of operations required as it eliminates the need to compute every possible combination of tensor products independently. Consequently, tensor network contraction transforms what would be an exponential problem in the brute-force method into a much more tractable one, providing a scalable and efficient way to reconstruct the quantum state in circuit-cutting scenarios, particularly those involving a large number of cuts.

Figure 4 provides an overview of the practical application of circuit cutting, showcasing its runtime across a diverse range of QC benchmarks. These benchmarks include tasks from quantum optimization algorithms and entangled states generation to key subroutines in quantum number factoring algorithms, all of which are pivotal in demonstrating the real-world applicability of QC. In these experiments, each benchmark circuit is constrained to a maximum of half the qubits and gates compared to its original, uncut counterpart, with the largest circuits tested on a 100-qubit QPU for benchmarks designed for up to 200 qubits. Notably, using tensor networks in circuit cutting (ScaleQC) is more than 1 billion times less classical post-processing overhead than brute force (CutQC).

This approach highlights the significant role of circuit cutting in enabling quantum computations that were previously unattainable because of hardware limitations. Without the use of circuit cutting, existing QC methods are restricted to executing quantum programs of considerably smaller scale. Moreover, the complexity and size of these benchmarks far exceed the capabilities of classical simulators; underlining the crucial enhancement that circuit cutting brings to the field of QC, particularly in bridging the gap between current quantum hardware limitations and the demands of advanced quantum algorithms.

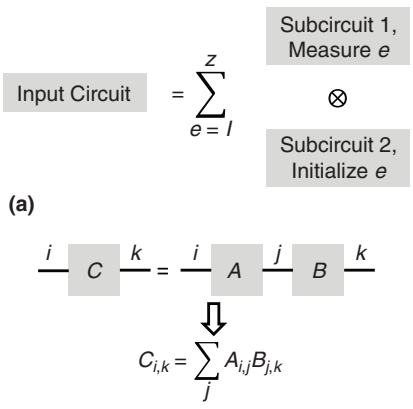


FIGURE 3. Reconstructing two subcircuits is equivalent to a pairwise tensor contraction. (a) Reconstructing a pair of subcircuits means multiplying and summing over the cut edges in between. (b) A pairwise tensor contraction with one shared index j, which is the inner dimension being contracted. i, k are the outer dimensions of the resulting big tensor C.

Industry acceptance

The industry's embrace of circuit-cutting technology is underscored by its integration into IBM's Qiskit software development kit, a toolkit used by more than half a million users globally. This significant move was further highlighted at IBM's 2022 annual quantum summit, showcasing the company's commitment to this innovative approach. Moreover, IBM has announced plans to develop its future quantum infrastructure around circuit cutting, signaling a major shift in the landscape of QC. The technology's potential and versatility have also attracted the attention of multiple companies, all actively exploring various use cases to leverage its capabilities. This widespread interest is further validated by the numerous grants and awards received, indicating a strong confidence in the practical applications and future prospects of circuit cutting in the QC industry.

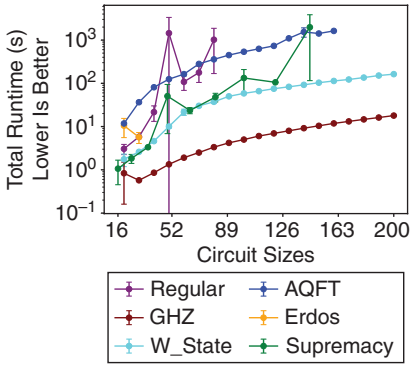


FIGURE 4. End-to-end wall clock runtimes, including cut searching, QPU runtime, and classical postprocessing. Each data point is the average of three trials. The error bars represent the standard deviations. Experiments terminate benchmarks when the estimated tensor contraction exceeds 10^{15} flops. AQFT: approximate quantum Fourier transform; GHZ: Greenberger–Horne–Zeilinger; Regular: 3-regular graphs; Erdos: Erdos–Renyi graphs; Supremacy: Google Quantum Supremacy experiment.

THE FUTURE OF DISTRIBUTED HYBRID COMPUTING

QC circuit cutting makes possible a novel hybrid CPU–QPU computing paradigm, fostering multidisciplinary collaborations and driving real-world applications far beyond mere proof of concept. The growing industry acceptance of these works underscores distributed hybrid computing's emergence as a pivotal aspect of QC. Looking ahead, there are exciting multidisciplinary opportunities to advance practical distributed hybrid computing. They include the following:

1. *Integrating classical high-performance computing techniques*: Bridging the gap between current quantum workloads and state-of-the-art distributed QC requires advancements in tensor network computing and the use of parallel GPUs, application-specified integrated circuits, and field-programmable gate arrays. For example, practical benchmarks may require between 10^{15} to 10^{20} flops of classical postprocessing. As a comparison, GPT-3 training requires about 10^{23} flops.
2. *Designing future hybrid QPU–CPU computing data centers*: The advent of cloud computing data centers for QC opens new avenues in distributed systems, such as optimizing for reduced latency and increased throughput. This involves tackling challenges in load balancing and resource allocation.
3. *Co-designing application and distributed hybrid CPU–QPU computing back ends*: Recognizing distributed QC as the standard back end for running workloads, domain experts across various fields are well positioned to develop more sophisticated and efficient algorithms, specifically optimized for these advanced computing platforms.
4. *Analyzing hybrid QPU–CPU computing complexity*: The future development of hybrid computing relies on a comprehensive theoretical understanding of its advantages and limitations beyond empirical evidence. Theory researchers should take the charge to study the complexities of hybrid systems and guide the development of efficient systems and applications.
5. *Addressing hybrid data security challenges*: Hybrid computing requires communications between quantum and classical back ends and hence may be susceptible to new data leakage channels. Hybrid data security will be the key to enabling trustworthy distributed hybrid computing.

In conclusion, distributed hybrid QPU–CPU computing represents a transformative path forward for QC, bridging the gap between current quantum hardware capabilities and the demands of advanced quantum algorithms. Circuit cutting represents the key methodology for making these distributed hybrid approaches possible. By enabling the execution of large quantum circuits on smaller scale QPUs, this technique not only makes quantum computations more feasible but also expands the range of solvable problems. The integration of classical computing and QC through this method underscores a pivotal shift toward practical, scalable, and efficient QC solutions, setting the stage for a future where complex quantum tasks are more accessible.

ACKNOWLEDGMENT

This article is based upon work supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, and Co-design Center for Quantum Advantage (C2QA) under contract DE-SC0012704. This work is funded in part by EPiQC, a National Science Foundation (NSF) Expedition in Computing, under award CCF-1730449. This article is based upon work supported by (while Martonosi was serving at) the NSF.

REFERENCES

1. I. L. Markov and Y. Shi, "Simulating quantum computation by contracting tensor networks," *SIAM J. Comput.*, vol. 38, no. 3, pp. 963–981, 2008, doi: 10.1137/050644756.
2. T. Peng, A. W. Harrow, M. Ozols, and X. Wu, "Simulating large quantum circuits on a small quantum computer," *Phys. Rev. Lett.*, vol. 125, no. 15, 2020, Art. no. 150504, doi: 10.1103/PhysRevLett.125.150504.
3. W. Tang and M. Martonosi, "ScaleQC: A scalable framework for hybrid computation on quantum and classical processors," 2022, *arXiv:2207.00933*.
4. W. Tang, T. Tomesh, M. Suchara, J. Larson, and M. Martonosi, "CutQC: Using small quantum computers for large quantum circuit evaluations," in *Proc. 26th ACM Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2021, pp. 473–486, doi: 10.1145/3445814.3446758.
5. W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, no. 5886, pp. 802–803, 1982, doi: 10.1038/299802a0.

WEI TANG is a final-year computer science Ph.D. student in Prof. Margaret Martonosi's group at Princeton University, Princeton, NJ 08540 USA. Contact him at weit@princeton.edu.

MARGARET MARTONOSI is the H.T. Adams '35 Professor of Computer Science at Princeton University, Princeton, NJ 08540 USA. She is a Fellow of IEEE and the Association for Computing Machinery. Contact her at mrm@cs.princeton.edu.